

IBM InfoSphere Information Server Deployment Architectures

Start small with a path for
massive growth

Understand metadata
considerations

Deploy in Grid and SMP
environments



Chuck Ballard
Tuvia Alon
Naveen Dronavalli
Stephen Jennings
Mark Lee
Sachiko Toratani



International Technical Support Organization

**IBM InfoSphere Information Server Deployment
Architectures**

December 2012

Note: Before using this information and the product it supports, read the information in “Notices” on page vii.

First Edition (December 2012)

This edition applies to Version 8.5 of IBM InfoSphere Information Server.

© Copyright International Business Machines Corporation 2012. All rights reserved.

Note to U.S. Government Users Restricted Rights -- Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Notices	vii
Trademarks	viii
Preface	ix
The team who wrote this book	xi
Now you can become a published author, too!	xiv
Comments welcome	xv
Stay connected to IBM Redbooks	xv
Chapter 1. InfoSphere Information Server overview	1
1.1 Introduction	2
1.2 Functional tiers	4
1.2.1 Services tier	5
1.2.2 Engine tier	6
1.2.3 Repository tier	7
1.3 Product modules	9
1.3.1 InfoSphere Blueprint Director	9
1.3.2 InfoSphere DataStage and InfoSphere QualityStage	10
1.3.3 InfoSphere Information Analyzer	11
1.3.4 InfoSphere FastTrack	12
1.3.5 InfoSphere Business Glossary	13
1.3.6 InfoSphere Metadata Workbench	14
1.3.7 InfoSphere Discovery	15
1.3.8 InfoSphere Data Architect	16
1.3.9 InfoSphere Information Server Manager, ISTOOL, and InfoSphere Metadata Asset Manager	17
Chapter 2. InfoSphere Information Server installation topologies	19
2.1 InfoSphere Information Server software tiers	20
2.2 Supported platforms	21
2.3 Single computer topology	23
2.4 Client server installation topology	24
2.5 Dedicated computer per tier topology	25
2.6 Dedicated engine topology	26
2.7 InfoSphere Information Server clustered installations	28
2.7.1 Active-passive cluster	28
2.7.2 Clustering the services tier	32
2.7.3 Clustering the InfoSphere Information Server repository tier	38
2.7.4 Engine tier clustering	43

2.7.5 Active-passive engine tier topology	47
2.7.6 Active-active engine tier topology	47
2.7.7 Massively parallel processing (MPP) topology	49
2.7.8 Parallel Engine Grid topology	52
2.7.9 Multiple dedicated engines topology	55
2.7.10 Web services engine	56
2.8 Disaster recovery	57
2.9 Chapter summary	59
Chapter 3. Metadata management	61
3.1 Defining metadata	62
3.2 Types of Metadata	63
3.2.1 Business metadata	63
3.2.2 Technical metadata	64
3.2.3 Operational metadata	65
3.2.4 Metadata usage classifications	65
3.3 Why metadata is important	66
3.3.1 Risk avoidance through trusted information	66
3.3.2 Regulatory compliance	67
3.3.3 IT productivity	67
3.3.4 Requirements for managing metadata	67
3.4 InfoSphere Information Server metadata	70
3.4.1 InfoSphere Information Server internal metadata	70
3.4.2 InfoSphere Information Server external metadata	70
3.4.3 Metadata lifecycle	70
3.5 Where to view InfoSphere Information Server metadata	71
3.5.1 Business requirements for deployment	71
3.5.2 Metadata environments	72
Chapter 4. Lifecycle management of assets	75
4.1 Introduction to assets	76
4.2 Introduction to source control and deployment	79
4.3 Source control and deployment in traditional software development	81
4.4 Source control and deployment in InfoSphere Information Server	84
4.5 Deployment scenarios and DataStage project structures	88
4.5.1 Method	88
4.5.2 Method summary	92
4.6 Source control and versioning with InfoSphere Information Server Manager and ClearCase	93
4.7 Deployment with InfoSphere Information Server Manager	95
4.8 Workflow for multiple release candidates	97
4.9 Context for deployment and versioning tools	100
4.10 Workflow for promotion into production	101

4.11	Build environment	102
4.12	Connecting with InfoSphere Information Server Manager to a domain	103
4.13	Installing the IBM Rational ClearTeam Explorer Eclipse plug-in	108
4.14	Integrating source control	114
4.15	Importing a source control project	122
4.16	Adding assets to Source Control Workspace	129
4.17	Checking out	136
4.18	Checking in	141
4.19	Refreshing the Workspace view	146
4.20	Creating and building deployment packages	149
4.21	Using the Deploy and Send options	153
4.22	Promoting deployment packages with ClearCase Explorer	155
4.22.1	Checking out deployment packages with ClearCase Explorer	156
4.22.2	Deploying a checked out package	158
4.23	Test data deployment	161
4.24	Lifecycle management of other assets	166
Chapter 5. Metadata deployment architectures		169
5.1	SIMPLE architecture	170
5.2	UNIFIED architecture	174
5.3	REPORTING architecture	185
5.4	GOVERNANCE architecture	191
5.5	Choosing which architecture to deploy	193
Chapter 6. Deployment use cases		195
6.1	Metadata differentiator	196
6.2	Sample use case scenarios	197
6.2.1	Scenario 1: Simple	198
6.2.2	Scenario 2: Mixed	204
6.2.3	Scenario 3: Complex	209
6.3	Summary	220
Glossary		223
Abbreviations and acronyms		229
Related publications		231
	IBM Redbooks	231
	Online resources	231
	Help from IBM	231

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM websites are provided for convenience only and do not in any manner serve as an endorsement of those websites. The materials at those websites are not part of the materials for this IBM product and use of those websites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

AIX®	IA®	Rational Team Concert™
ClearCase®	IBM®	Rational®
Cognos®	IMS™	Redbooks®
DataStage®	Information Agenda®	Redbooks (logo)  ®
DB2 Universal Database™	Informix®	Tivoli®
DB2®	InfoSphere®	WebSphere®
GPFS™	Optim™	z/OS®
Guardium®	PowerHA®	
HACMP™	QualityStage®	

The following terms are trademarks of other companies:

Netezza, and N logo are trademarks or registered trademarks of IBM International Group B.V., an IBM Company.

LSF, and LSF ACTIVECLUSTER are trademarks or registered trademarks of Platform Computing, an IBM Company.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java, and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Preface

IBM® InfoSphere® Information Server provides a comprehensive, metadata-driven platform for delivering trusted information across heterogeneous systems. InfoSphere Information Server has a scalable, parallel engine that facilitates data profiling, data integration and data delivery through reusable, shared components (such as connectors, data quality, complex transformations) and rules that use a common metadata platform.

InfoSphere Information Server has a unique, metadata-driven design that helps to align business goals and IT activities. It provides consistent terms and rules, and captures business specifications. It uses these specifications to automate development tasks, helping you gain greater insight to data by tracking its lineage.

Typical deployment architectures introduce challenges to fully using the shared metadata platform across products, environments, and servers. And, data privacy and information security requirements add even further levels of complexity.

This IBM Redbooks® publication presents guidelines and criteria for the successful deployment of InfoSphere Information Server components in typical logical infrastructure topologies that use the shared metadata capabilities of the platform. These guidelines support requirements for development lifecycle, data privacy, information security, high availability, and performance. The objective of this publication is to help you evaluate your own information requirements to determine an appropriate deployment architecture, based on those guidelines. It can help you fulfill specific use cases, and help you effectively use the functionality of your InfoSphere Information Server product modules and components to successfully achieve business goals.

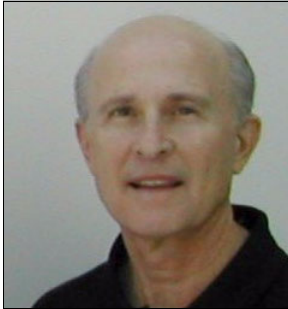
This book is intended for IT architects, information management and information integration specialists, and system administrators who are responsible for delivering the full suite of information integration capabilities that are provided by IBM InfoSphere Information Server.

This book is organized into the following chapters:

- ▶ Chapter 1: InfoSphere Information Server overview. This chapter has an overview of the architecture of IBM InfoSphere Information Server, and its functional and product components.
- ▶ Chapter 2: InfoSphere Information Server installation topologies. This chapter describes the installation topologies for InfoSphere Information Server, ranging from basic single computer through to highly available clusters and grid configurations.
- ▶ Chapter 3: Metadata management. Metadata management refers to the tools, processes, and environment that can help an organization answer the question, “How do we know what we know about our data?” This chapter helps you define metadata, understand the types of metadata and why metadata is important, and where to view InfoSphere Information Server metadata.
- ▶ Chapter 4: Lifecycle management of assets. This chapter describes methodologies to implement lifecycle management, including source code control management for IBM InfoSphere DataStage® and IBM QualityStage® applications using IBM InfoSphere Information Server Manager and external source code control systems. In addition, this chapter describes the lifecycle management of IBM InfoSphere Information Analyzer, IBM InfoSphere Business Glossary, IBM InfoSphere Metadata Workbench, and related Metadata Deployment Architectures.
- ▶ Chapter 5: Metadata deployment architectures. To effectively use the InfoSphere Information Server product modules and achieve the business objectives, you must consider various metrics when you choose the deployment architecture for InfoSphere Information Server. This chapter introduces the following deployment architectures: SIMPLE, UNIFIED, REPORTING, and GOVERNANCE.
- ▶ Chapter 6: Deployment use cases. This chapter documents a number of use cases to test and validate the InfoSphere Information Server deployment architectures.

The team who wrote this book

This book was produced by a team of specialists from around the world working with the International Technical Support Organization, in San Jose, California.



Chuck Ballard is a Project Manager at the International Technical Support organization, in San Jose, California. He has over 35 years of experience, holding positions in the areas of product engineering, sales, marketing, technical support, and management. His expertise is in the areas of database, data management, data warehousing, business intelligence, and process re-engineering. He writes extensively about these subjects, teaches classes, and presents at conferences and seminars worldwide. Chuck has both a Bachelor's degree and a Master's degree in Industrial Engineering from Purdue University.



Tuvia Alon is a Senior Consultant and Solution Architect in IBM Software Group Lab Services Center of Excellence in Information Management, servicing EMEA and other regions worldwide, operating from the IBM Israel Software Labs. Tuvia joined IBM in 2006 with the acquisition of Unicorn, which specializes in metadata and semantic technologies. Tuvia has over 20 years of experience in developing, implementing, and supporting enterprise software. He is an expert in metadata technologies and a recognized leader in implementing enterprise data governance solutions. As a Consultant in the Center of Excellence, Tuvia has served as a trusted advisor for IBM customers who are implementing data integration and data governance strategies. He is a presenter at numerous conferences in the US and Europe about the use of metadata in corporate governance policy. He recently co-authored the IBM Redbooks *Metadata Management with IBM InfoSphere Information Server*, SG24-7939.



Naveen Dronavalli is a Technical Solution Architect with the Virtual Services team, in the IBM Information Management group. He has over eight years of experience in the information technology field, specializing in data warehousing and data integration. Naveen is skilled in tools, such as the IBM InfoSphere Information Server Suite, databases, and operating systems. He held technical lead positions in architecture design of ETL and data integration processes, and provides infrastructure solution architectures for InfoSphere Information Server on Grid that uses high availability. He also provides guidance, mentoring, and training for clients. Naveen has a Bachelor's degree in Electronics and Communication Engineering from Kuvempu University (India), and a Master's degree in Electrical Engineering with a major in Computer Engineering from Illinois Institute of Technology.



Stephen Jennings is an IBM Client Technical Specialist in Information Management and was previously an IBM Technical Solution Architect. He has over 10 years of experience helping customers with data integration and data governance projects using InfoSphere Information Server. Steve is a graduate of the University of Louisville and is located in Louisville, Kentucky, US.



Mark Lee is a Senior Technical Consultant within the IBM Software Group. He has worked with a wide range of IBM customers helping them implement complex data integration and parallel processing solutions over a period of 20 years. Mark has an Honors degree and a Master's degree in Computer Science from the University of Manchester. He is located in Surrey, England.



Sachiko Toratani is an IT Specialist for IBM InfoSphere products at IBM Japan. She has over 10 years of experience in DBMS and application development in government-related systems. Her areas of expertise include Information Integration and DBMS, with extensive skills in InfoSphere Information Server and DataStage. She is IBM certified in Database Administrator IBM DB2® Universal Database™ for Linux, UNIX, and Windows. She also contributed to the development of IBM Redbooks *IBM InfoSphere DataStage Data Flow and Job Design*, SG24-7576 and *Deploying a Grid Solution with the IBM InfoSphere Information Server*, SG24-7625.

Other contributors

We want to thank others who contributed to this book, in the form of written content, subject expertise, and support.

A special thanks for being the technical experts on this project:

- ▶ Paul Christensen: IBM Information Agenda® Architect, IBM Software Group, Information Management, West Chester, PA, US
- ▶ Robert Johnston: Information Agenda Architect, IBM Software Group, Information Management, Cleveland, OH, US

A special thanks for content contribution:

- ▶ Julius Lerm: Advanced Consulting Engineer and Technical Solution Architect, IBM Software Group, Information Management, Chicago, IL, US

A special thanks for their contributions to this project:

- ▶ Tony Curcio: InfoSphere Product Management, Software Product Manager, Charlotte, NC, US
- ▶ Marc Haber: Functional Architect, Metadata Tools, IBM Software Group, Information Management, Haifa, Israel
- ▶ Ernie Ostic: Client Technical Specialist, IBM Software Group, Worldwide Sales Enablement, Piscataway, NJ, US

From the International Technical Support Organization:

- ▶ Mary Comianos: Publications Management
- ▶ Emma Jacobs: Graphics Support
- ▶ Ann Lund: Residency Administration
- ▶ Diane Sherman: Editor

Now you can become a published author, too!

Here's an opportunity to spotlight your skills, grow your career, and become a published author—all at the same time! Join an ITSO residency project and help write a book in your area of expertise, while honing your experience using leading-edge technologies. Your efforts will help to increase product acceptance and customer satisfaction, as you expand your network of technical contacts and relationships. Residencies run from two to six weeks in length, and you can participate either in person or as a remote resident working from your home base.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our books to be as helpful as possible. Send us your comments about this book or other IBM Redbooks publications in one of the following ways:

- ▶ Use the online **Contact us** review Redbooks form found at:
ibm.com/redbooks
- ▶ Send your comments in an email to:
redbooks@us.ibm.com
- ▶ Mail your comments to:
IBM Corporation, International Technical Support Organization
Dept. HYTD Mail Station P099
2455 South Road
Poughkeepsie, NY 12601-5400

Stay connected to IBM Redbooks

- ▶ Find us on Facebook:
<http://www.facebook.com/IBMRedbooks>
- ▶ Follow us on Twitter:
<http://twitter.com/ibmredbooks>
- ▶ Look for us on LinkedIn:
<http://www.linkedin.com/groups?home=&gid=2130806>
- ▶ Explore new Redbooks publications, residencies, and workshops with the IBM Redbooks weekly newsletter:
<https://www.redbooks.ibm.com/Redbooks.nsf/subscribe?OpenForm>
- ▶ Stay current on recent Redbooks publications with RSS Feeds:
<http://www.redbooks.ibm.com/rss.html>



InfoSphere Information Server overview

IBM InfoSphere Information Server is a product family that provides a unified data integration platform, enabling customers to understand, cleanse, transform and deliver trustworthy and context-rich information to critical business initiatives. InfoSphere Information Server includes a variety of product modules and components that provide an integrated end-to-end information integration solution, including the facility for governance throughout the process. A key element of the InfoSphere Information Server platform is a rich, shared metadata layer that provides visibility into both the business and the technical realms, a foundation for reporting, and the integration points across product modules.

InfoSphere Information Server provides a flexible architecture that allows organizations to maximize the benefits of the platform, to achieve their business objectives. Determining a successful product module deployment strategy requires consideration of many parameters, in a complex information integration set up.

The ultimate objective of this book is to help you evaluate your own information requirements. It can help you determine an appropriate deployment architecture, based on the guidelines presented, that can fulfill your specific use cases, while you effectively use the functionality of your InfoSphere Information Server product modules and components to successfully achieve business goals. To that end, this book presents typical lifecycle processes for development product

modules that are included in InfoSphere Information Server, describes how the development code is managed and promoted through the various development phases, and provides multiple architectural infrastructures that support these environments.

However, code is not the end-product that is delivered to the business in an information integration project. Rather, it is the information that results from extracting, transforming, loading (ETL), and the cleansing and enrichment processes, which improve the speed, accuracy, and value of the decisions that can be made with that information. This information consists of both data and the metadata that describes and is related to it.

Data management is a mature field of study with much written about it, including references in this book. However, this book also presents how metadata is generated by the InfoSphere Information Server product modules and enables integration between them, how metadata is managed differently from development assets and therefore, how the architectures should be designed to maximize the benefits of the metadata.

The book begins with a general introduction of the InfoSphere Information Server through a description of its components. The following topics are presented in this chapter:

- ▶ Introduction and functional overview of InfoSphere Information Server
- ▶ Functional tier infrastructure of InfoSphere Information Server platform
- ▶ Product modules and components of InfoSphere Information Server

Version 8.7: Most of the methodologies, processes, and architectural descriptions are applicable to separate versions of InfoSphere Information Server, unless otherwise mentioned. InfoSphere Information Server version 8.7 is used as a baseline.

1.1 Introduction

IBM InfoSphere Information Server provides a single unified platform that enables companies to understand, cleanse, transform, and deliver trustworthy and context-rich information. It is built on a multitiered platform that includes a suite of product modules and components, primarily focusing on aspects of the information integration domain. Furthermore, it integrates with other third-party applications to use information, where it exists in the enterprise.

InfoSphere Information Server supports a wide range of initiatives, including business intelligence, master data management, infrastructure rationalization, business transformation, and risk and compliance (governance).

Business intelligence

Developing a unified view of the business for better decisions can be more easily accomplished with InfoSphere Information Server. It helps in understanding existing data sources, cleansing, correcting and standardizing information, and also loading analytical views that can be reused throughout the enterprise.

Master data management

InfoSphere Information Server simplifies the development of authoritative master data by showing where and how information is stored across source systems. It also consolidates disparate data into a single, reliable record, cleanses and standardizes information, removes duplicates, and links records across systems. This master record can be loaded into operational data stores, data warehouses, or master data applications. The record can also be assembled, completely or partially, on demand.

Infrastructure rationalization

InfoSphere Information Server aids in reducing operating costs by showing relationships among systems and by defining migration rules to consolidate instances or move data from obsolete systems. Data cleansing and matching ensure high-quality data in the new system.

Business transformation

InfoSphere Information Server can speed development and increase business agility by providing reusable information services that can be plugged into applications, business processes, and portals. These standards-based information services are maintained centrally by information specialists, but are widely accessible throughout the enterprise.

Risk and compliance (governance)

InfoSphere Information Server helps improve visibility and data governance by enabling complete, authoritative views of information with proof of lineage and quality. These views can be made widely available and reusable as shared services; the rules inherent in them are maintained centrally.

1.2 Functional tiers

InfoSphere Information Server consists of a robust, scalable, server architecture that is built on the following three distinct components, and thin and rich client product modules (client tier), shown in Figure 1-1:

- ▶ J2EE application server (services tier)
- ▶ Database (repository tier)
- ▶ Parallel processing runtime framework (engine tier)

Both the application server and the database server are standard server applications. Most enterprises already have the skills to manage and administer these server applications, particularly because the InfoSphere Information Server infrastructure is designed for minimal intervention.

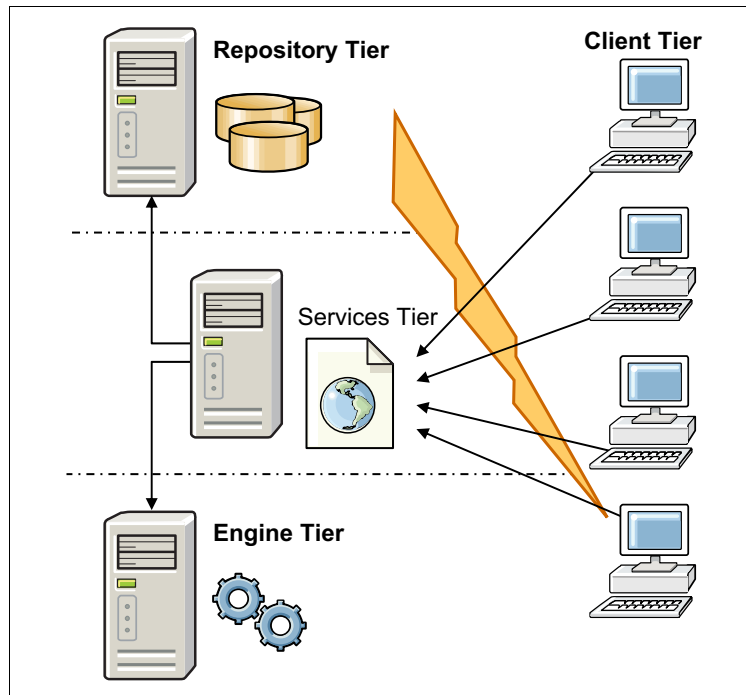


Figure 1-1 InfoSphere Information Server high-level architecture

One of the key advantages of InfoSphere Information Server is this shared infrastructure for all the individual InfoSphere Information Server product modules. This shared infrastructure enables integration of all these components, thereby, minimizing duplicate effort and maximizing reuse. In addition, InfoSphere Information Server integrates metadata from external applications

into its repository, enabling the InfoSphere Information Server product modules to use this external metadata to achieve business objectives.

1.2.1 Services tier

The InfoSphere Information Server services tier is built on IBM WebSphere® Application Server. WebSphere Application Server provides the infrastructure for the common services across all modules, such as authentication and repository access, and also services and web applications that are proprietary to the individual product modules and components. Figure 1-2 shows some of the product module-specific applications and services, and also some of the common services that are shared across all of the product modules.

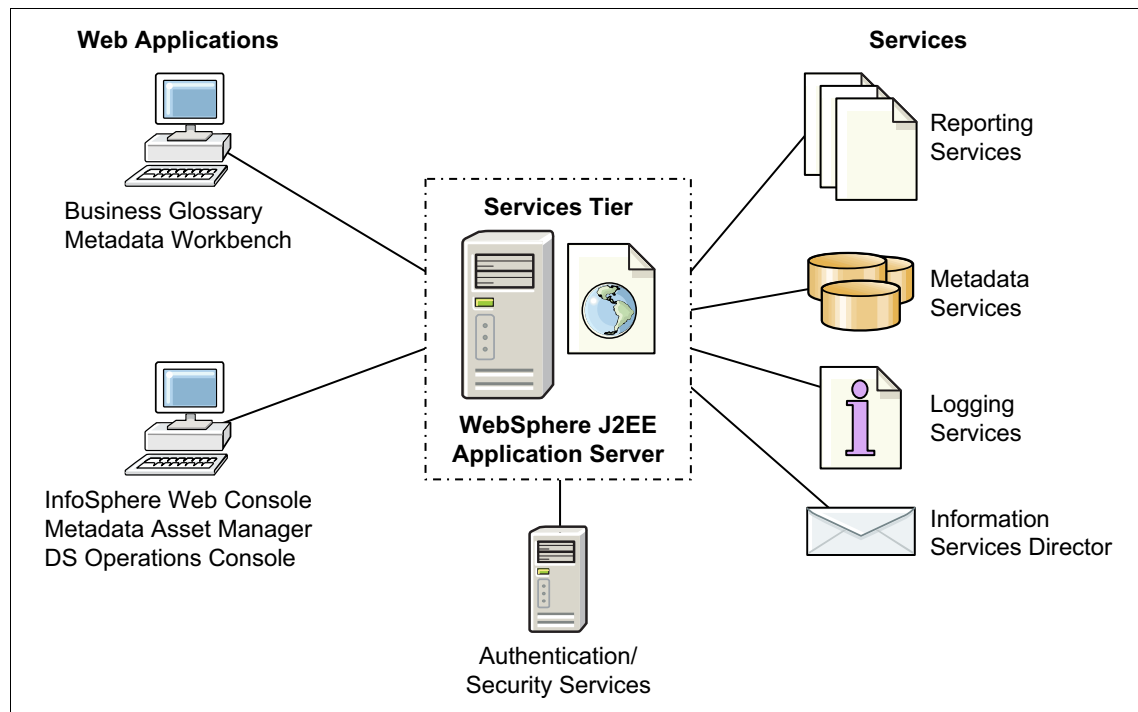


Figure 1-2 InfoSphere Information Server services tier

Security is one of the major functions managed by WebSphere Application Server. When InfoSphere Information Server is first installed, WebSphere Application Server automatically configures the InfoSphere Information Server internal user registry (persisted in the InfoSphere Information Server repository) as the default user registry. Afterward, it is possible to reconfigure WebSphere Application Server to authenticate through a different user registry such as the

operating system or LDAP (Active Directory, for example), to simplify the administration of InfoSphere Information Server users and groups. This area is one of two areas where change the default WebSphere Application Server configuration might be necessary. When a larger load is on the services tier, it might also be necessary to increase the maximum heap size of the WebSphere Application Server Java virtual machine (JVM), dedicated to InfoSphere Information Server use.

No matter which authentication method is used, each of the product modules and components uses WebSphere Application Server to authenticate. In this manner, only one authentication mechanism is required for the entire InfoSphere Information Server platform and is used by each of the product modules and components. As a result, WebSphere Application Server is able to maintain session and log activity for each active product module (session).

Each product module uses the repository for its own persistence layer, yet its only access to the repository is through the WebSphere Application Server repository service. WebSphere Application Server is configured at installation time to access the repository through its JDBC layer, based on the credentials passed to it during the installation. This is one way that InfoSphere Information Server ensures the integrity and security of its data and metadata.

1.2.2 Engine tier

The InfoSphere Information Server engine tier provides parallel processing runtime functionality for several InfoSphere Information Server product modules. This functionality is the basis for virtually unlimited scalability, because the only limit to processing capability is the available hardware. It is important to note that the parallel framework can take advantage of all of the hardware resources that are available on the engine tier server platform. As such, in a high-load processing scenario, a strong suggestion is for the engine tier to have its own hardware resources that it does not share with the other tiers.

The engine tier, as shown in Figure 1-3 on page 7, includes built-in native drivers for data connectivity to external data sources, whether for data cleansing, profiling, quality analytics, or transformation, that take advantage of better performance and functionality. In addition, it is possible to use the ODBC drivers that are included with InfoSphere Information Server, or other external drivers, for connectivity to many other data sources. Because processing millions of records

is often necessary, the parallel processing engine provides an efficient method for accessing and manipulating various kinds of data.

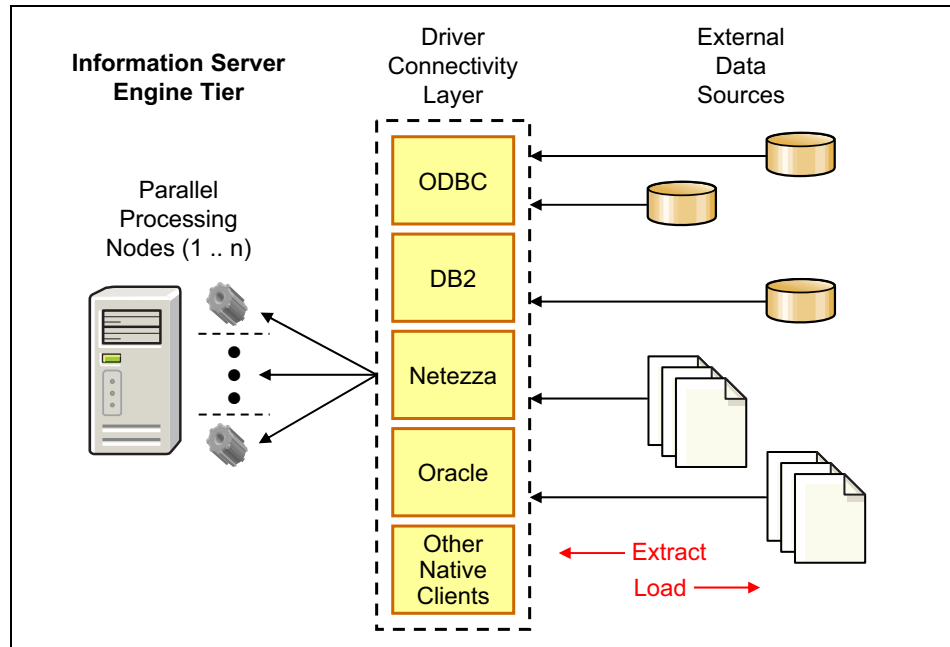


Figure 1-3 InfoSphere Information Server engine tier

1.2.3 Repository tier

The InfoSphere Information Server repository tier is built on a standard RDBMS (IBM DB2, Oracle, or Microsoft SQL Server). The repository tier provides the persistence layer for all of the InfoSphere Information Server product modules and components.

The InfoSphere Information Server *repository* (in the repository tier) is a single repository where information about data source, data integration jobs, business glossary, data warehouse and data mart, reports, and report models are all brought together into one shared location, which can be used by InfoSphere Information Server product modules and components within the suite.

In addition, to ensure data integrity and processing performance, and to provide temporary persistence, two InfoSphere Information Server product modules (IBM InfoSphere QualityStage and IBM InfoSphere Information Analyzer) also use their own schemas as a workspace (see Figure 1-4 on page 8). When the work is done, the relevant metadata in the workspace is published to the shared InfoSphere Information Server repository, at user-designated intervals, to be

used by other product modules. Furthermore, other operations can be used for collecting and persisting run time, performance, and logging data that is used for monitoring jobs and evaluating system efficiency.

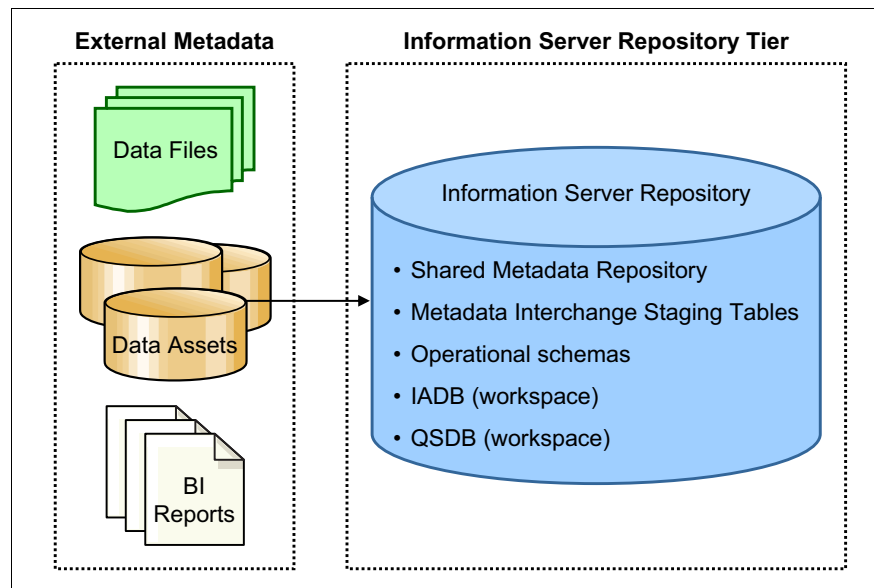


Figure 1-4 InfoSphere Information Server repository tier

As described previously, each of the product modules persists its own data in the InfoSphere Information Server repository. Therefore, much of the metadata in the repository is internally generated data. However, also possible is to import, load, and append metadata from external applications, such as data modelling tools, business intelligence applications, and other sources of structured data, that have relevance to one or more InfoSphere Information Server product modules or features. InfoSphere Information Server has a feature for users to import this third-party metadata into a staging area. This staging area provides an interim load area for external metadata, where this new metadata can be compared with existing metadata (before loading) to assess the impact and act on it, and also keep a record of the loaded metadata.

In cases where an external application does not provide access to its metadata (either directly or through extracts), documenting these relevant external information assets in the repository, through controlled, manual processes and tools, is also possible.

1.3 Product modules

InfoSphere Information Server offers a collection of product modules and components that work together to achieve business objectives within the information integration domain. The product modules are designed to provide business and technical functionality throughout the entire initiative from planning through design to implementation and reporting phases.

InfoSphere Information Server consists of the following product modules and components: IBM InfoSphere Blueprint Director, IBM InfoSphere DataStage, IBM InfoSphere QualityStage, IBM InfoSphere Information Analyzer, IBM InfoSphere FastTrack, IBM InfoSphere Business Glossary, and IBM InfoSphere Metadata Workbench, and also IBM InfoSphere Data Architect and IBM InfoSphere Discovery, which are integrally related applications. In addition to the product modules, InfoSphere Information Server includes the following utilities for importing external assets into the repository, deploying internal assets from one environment to another, and also managing these assets to remove duplicates and orphans:

- ▶ InfoSphere Information Server Manager
- ▶ The istool command-line utility (CLI)
- ▶ IBM InfoSphere Metadata Asset Manager

This section describes each of these product modules and utilities, including their general functionality, the metadata that is associated with them, and the infrastructure components on which they are based.

1.3.1 InfoSphere Blueprint Director

InfoSphere Blueprint Director is a graphical design tool that is used primarily for creating high-level plans for an InfoSphere Information Server based initiative, whether in information governance, information integration, data quality, business intelligence, or any other kind of information-based project. To simplify the task, Blueprint Director is bundled with several ready-to-use, project-type-based content templates that can be easily customized to fit the project. Alternatively, a new blueprint can be created as needed.

Blueprint Director has a design canvas onto which standard graphical objects, representing processes, tasks or anything else, are dragged and dropped. Objects can be connected, one to the other, thereby implying a sequential order to these events or dependencies between them. Each graphical object has a label that indicates its purpose; however, the object can optionally be linked to content that was produced and published in IBM Rational® Method Composer, for greater detail. When a single object represents a number of tasks or

processes, the object can drill down to create or link to a more detailed layer of the blueprint diagram, so that the final blueprint is likely to contain several hierarchical levels of processes (and subprocesses). The hierarchical blueprint diagram, combined with the methods (text descriptions), forms the basis of the project plan as built from top to bottom (high to low level).

Blueprint Director is a unique component among the InfoSphere Information Server product modules and components, in that it is a stand-alone, Eclipse-based, client-only application that does not have any dependencies on the InfoSphere Information Server infrastructure for persistence, authentication, or other shared services. This way provides a useful flexibility for planning the project at an early stage before all of the infrastructure is ready and available. However, it is expected that in InfoSphere Information Server version 9.1, it will be possible to publish a blueprint (read-only) to InfoSphere Information Server that can be viewed through Metadata Workbench (MWB) or Business Glossary (BG) is expected to be possible.

Blueprint Director, V8.5 on which this book is based, does not generate any metadata that is currently consumed by any of the InfoSphere Information Server product modules and components, but the newer V9.1 does. However, V8.5 has its own persistence layer in an XML format (*.bpd) file. And, V8.5 has the facility to link to InfoSphere Information Server repository objects, and even launch InfoSphere Information Server product modules and components to display these linked objects in their native tool.

1.3.2 InfoSphere DataStage and InfoSphere QualityStage

InfoSphere DataStage and InfoSphere QualityStage provide essential integration functionality for a wide range of projects. DataStage integrates data across multiple source and target applications, collecting, transforming, and delivering high volumes of data. QualityStage provides data cleansing functionality from standardization, to deduplication, to establishing master data. Both product modules can take advantage of the parallel processing runtime engine (on the engine tier) to run their jobs efficiently, in a scalable manner that optimizes data flow and available system resources.

In general, both DataStage and QualityStage jobs are launched as batch operations (using the DataStage and QualityStage Director, command line, or enterprise scheduler, for example). However, enabling these jobs to constantly run as a service (in real time) is possible by using the IBM InfoSphere Services Director (ISD), while still using the parallel processing runtime engine. In an “always run” scenario, the jobs are deployed by using ISD on the WebSphere Application Server services bus, and further by using the benefits of the InfoSphere Information Server services tier.

DataStage and QualityStage share the same rich user interface, the DataStage and QualityStage Designer, which provides a straightforward method to design jobs that extract, transform, and load (ETL):

- ▶ *Extract* data from various data sources
- ▶ *Transform* (or process) the data according to the business requirements
- ▶ *Load* the resultant data into the relevant target data storage systems

DataStage and QualityStage also share two additional rich client applications for administering projects and also testing, executing, and troubleshooting jobs:

- ▶ DataStage and QualityStage Administrator
- ▶ DataStage and QualityStage Director

In addition, they share a web application, referred to as the DataStage and QualityStage Operations Console, running on the services tier with agents on the engine tier, that provides additional runtime job and system monitoring and performance metrics.

DataStage and QualityStage projects are persisted in the InfoSphere Information Server repository. This portion of the repository storage model contains DataStage and QualityStage job designs and other project artifacts such as table definitions, containers, parameter sets, connector objects, and more. DataStage and QualityStage design, and optionally, runtime operational metadata, are key components in calculating impact and lineage reports, based on accumulated metadata in the repository. Besides generating design and operational metadata, DataStage and QualityStage jobs consume physical metadata for populating table definitions, used for designing DataStage and QualityStage stage links within their jobs.

1.3.3 InfoSphere Information Analyzer

IBM InfoSphere Information Analyzer (IA®) helps assess and monitor data quality, identify data quality concerns, demonstrate compliance, and maintain audit trails. The IA Workbench is a rich client, providing a graphical user interface that performs two functions: quality assessment (with data rules) and data profiling (with column analysis). Information Analyzer requires that the metadata from these data sources reside in the repository, including the connectivity information (such as the driver that is used to connect to the source and the DSN and URL), before performing these activities. Importing this technical metadata can be done from InfoSphere Information Analyzer, or from an InfoSphere Information Server utility named InfoSphere Metadata Asset Manager (see 1.3.9, “InfoSphere Information Server Manager, ISTOOL, and InfoSphere Metadata Asset Manager” on page 17).

Information Analyzer uses the connectivity and the parallel processing functionality of the parallel engine, on the engine tier, to query the data sources and load the column values in the Information Analyzer database workspace (IADB) on the repository tier. Information Analyzer then does the profiling analysis of each selected column separately. All of the analysis results are persisted in the IADB workspace until it is refreshed by rerunning the process; however, a baseline can also be established for comparison. All of the processing is coordinated through proprietary services running on the WebSphere Application Server services tier. When the work is complete, you can publish the analysis results from the IADB workspace to the shared repository, where it can be used by the other product modules and components (particularly DataStage and FastTrack).

The manner in which Information Analyzer assesses the level of data quality is by establishing constraints (data rules) to which the data should (or should not) adhere, and then test whether the data complies with these rules. This functionality serves on-going monitoring of the data (trend analysis), which is often part of a data quality and an information governance initiative.

1.3.4 InfoSphere FastTrack

InfoSphere FastTrack is a rich client that is designed to create source-target (or technically target-source) mapping specifications to be used by developers of data integration jobs. The main panel of the client application provides a spreadsheet-like, columnar structure for entering (copying or dragging) objects from the repository display into cells for source columns, target columns, and assigned business terms. It also provides manual descriptions of source to target transformations, and optionally, DataStage Transformer stage-specific code.

The completed mapping specification can be output in several formats, including an annotated DataStage job generated directly by FastTrack. This format is particularly useful for the DataStage developer because the specification is delivered in a manner in which the developer is familiar. In addition, this delivery format provides a job template that can be used as the basis for creating a new job, including some design artifacts that can be copied to the new job as-is.

Additional advantages of using FastTrack for mapping specification documentation include features:

- ▶ Centrally stored and managed specifications
- ▶ Simple drag and drop functionality for specifying source and target columns
- ▶ Accuracy of source and target column names (that actually exist in the repository) with assured correct spelling

- ▶ Visibility of metadata, such as data types, for each of the source and target columns, which can be helpful in determining appropriate mapping relationships and transformations
- ▶ “Discover” mappings, joins, and lookups assistance, based on published data profiling results, name recognition, and business term assignment
- ▶ Persistence layer in shared repository can be used in lineage reports

1.3.5 InfoSphere Business Glossary

InfoSphere Business Glossary primarily uses a thin client browser approach, but also several other interfaces, to help users share information across the organization. The basic content is a glossary of terms with their definitions, organized in categories that provide containment, reference, and context. Both terms and categories include several descriptive attributes, and also other attributes that define relationships to other InfoSphere Information Server repository objects (both technical and business related), information stewards, development cycle statuses, and even optional user-created custom attributes.

After the terms are published to the glossary, they are accessible to users through various search and browse features, available in the following interfaces:

- ▶ Business Glossary (web browser client)
- ▶ Business Glossary Anywhere pop-up client (rich client on workstation)
- ▶ IBM Cognos® context sensitive glossary search and display (menu option)
- ▶ Eclipse plug-in (uses REST API) for InfoSphere Data Architect and other Eclipse-based products
- ▶ REST API programmable access through custom programming

In addition to the search and browse functions that are available in the Business Glossary related interfaces, Information Analyzer and FastTrack provide specific views of the business metadata contained in Business Glossary. InfoSphere Metadata Workbench provides specific views of the business metadata too, but also provides for custom metadata queries that can certainly include the business metadata from Business Glossary.

Business Glossary browser provides the only graphical user interface for authoring (a read and write interface for creating, modifying, and deleting terms and categories) the glossary, and is the main tool for searching and browsing (read only). The available functionality is based on roles, as defined by the InfoSphere Information Server administrator. However, an additional, finer-grained, access control exists and is managed by anyone with the Business Glossary administrator role.

Although Business Glossary provides a functional web browser interface for creating and modifying the business glossary, an administrator can bulk-load pre-existing glossary information from appropriately formatted CSV and XML files. This way provides a useful facility for using glossary information that might be contained in external applications or spreadsheets. Furthermore, there is an API (REST) that can be accessed programmatically, through custom application development, for retrieving existing content or modifying the content.

Business Glossary persists all of its data (business metadata) in the InfoSphere Information Server repository. This approach is particularly appropriate because business terms might have relationships with physical assets (such as tables, columns and report headers, to name a few) that already exist in the repository. Furthermore, in addition to creating or modifying the glossary, Business Glossary can be used to both view the published business metadata and view other metadata objects in the repository, similar to InfoSphere Metadata Workbench, but from a different view perspective.

1.3.6 InfoSphere Metadata Workbench

The primary function of InfoSphere Metadata Workbench is to provide a “view” of the content of the InfoSphere Information Server repository, so that users can get answers to questions about the origin, history, and ownership of specific assets. This “view” is available in several formats:

- ▶ Three pre-configured Metadata Workbench reports: impact analysis, business lineage (also available in Business Glossary), and data lineage. These metadata-based reports highlight the relationships and connections across metadata objects, and the potential impact of changes to any object in the chain of metadata.
- ▶ Simple search, and complex “ad hoc” and persisted custom queries. Here, the user specifies a list of metadata attributes to display for any given set of constraints.
- ▶ Browse feature that can display any category of metadata assets (such as hosts, tables, columns, terms and BI reports, and also the other metadata objects), which provides the user an opportunity to select the specific instance of the metadata for a “drill-down” display.

All of these reports are displayed on-screen, but most can also be exported to CSV-formatted files for external use and further analysis.

In addition to the repository content views and reports, Metadata Workbench can enhance existing metadata with descriptions and custom attributes, and with assigning related metadata assets, such as other database metadata, business terms, model objects, and even data stewards (that is, read/write functionality).

Furthermore, Metadata Workbench can “create” extended lineage objects for the purpose of adding metadata to the repository that is not generated automatically by InfoSphere Information Server product modules and components, or that cannot be easily imported from external applications. This approach provides both documentation of external processes, and enables extending a business or data lineage report beyond the boundaries of the client’s use of InfoSphere Information Server.

Rendering Data Lineage Reports or invoking Queries (ad-hoc reports) from the Workbench, specifically those that might analyze and report on many hundreds of information assets, can require additional system resources and configuration within the Metadata Workbench. Such actions use the available resources of the services tier, which is shared across the components of the InfoSphere Information Server. The Metadata Workbench, by design, limits the results of such actions according to the system configuration, to prevent overloading the services tier. Therefore, enhancing the system configuration to match the actual deployed configuration of the services tier might be required.

1.3.7 InfoSphere Discovery

InfoSphere Discovery is an automated data relationship discovery solution that helps you to understand the data content, relationships, and transformations, to discover business objects, and to identify sensitive data within and across multiple heterogeneous data stores. The automated results derived from Discovery are actionable, accurate, and easy to generate, especially when compared to manual (non-automated) data analysis approaches that many organizations still use today.

Discovery works by *automatically* analyzing data sources and generating hypotheses about the data. Throughout the process, it interrogates the data, and generates metadata that includes a *data profile*, or *column analysis*.

In the context of an information integration project, this metadata provides an understanding of data and their relationships. It can be used for governance, or simply to help in the source-to-target mapping, as a planning aid to ETL specification.

Discovery directly accesses the source data systems to get the data, for data relationship discovery purposes. When the results are ready to be shared in the InfoSphere Information Server repository, usually for Business Glossary, FastTrack or both, its export and import utility can publish the results to the shared InfoSphere Information Server repository. These results enhance the physical metadata, associate business terms with physical assets, and also assist in designing mapping specifications for ETL, based on data content and relationships across disparate data stores.

1.3.8 InfoSphere Data Architect

InfoSphere Data Architect is an enterprise data modeling and integration design tool. It can be used to discover, model, visualize, relate, and standardize diverse and distributed data assets. It is structurally similar to Blueprint Director, in that it is a stand-alone, Eclipse-based, client-only product module with its own persistence layer (XML format files *.dbm, *.l dm, *.ndm, *.ddm), potentially with cross-file links (relationships) between the various models.

From a top-down approach, Data Architect can be used to design a logical model and automatically generate a physical data model from the logical source. DDL scripts can be generated from the physical data model to create a database schema based on its design. Another possibility is for Data Architect to connect to the RDBMS and instantiate the database schema directly from the Data Architect physical data model. This “generation” facility works both ways in that it is also possible to “reverse engineer” an existing, instantiated database, into a Data Architect physical data model for modification, reuse, versioning and other purposes.

Rather than designing models from the beginning, a possible approach is to purchase one of the IBM Industry Models in a format that is consumable by Data Architect. In this manner, it is possible to jump-start the database design phase of the project and benefit from rich data modeling expertise of IBM in the specific industry. Standard practice is to scope the industry standard logical model to also fit the customer's requirements, and build an appropriate physical data model that combines industry standards with customer specifics. An added advantage of the IBM Industry Models package for InfoSphere Data Architect is that it includes an Industry standard glossary model that can be used to populate the Business Glossary, complete with relationships (assigned assets) to the Data Architect logical model.

Data Architect does not rely on the InfoSphere Information Server infrastructure for any shared services or persistence. However, Data Architect models (logical, data, and glossary) can be imported into the InfoSphere Information Server repository by using the InfoSphere Metadata Asset Manager utility, described in 1.3.9, “InfoSphere Information Server Manager, ISTOOL, and InfoSphere Metadata Asset Manager” on page 17. Furthermore, Business Glossary terms and categories can be associated directly with Data Architect logical model entities and attributes, and also data model tables and columns, with “drag and drop” facility by using the Business Glossary Eclipse (BGE) plug-in for Data Architect. With BGE, Business Glossary is downloaded to an offline XML format file for use with Data Architect. This offline glossary is manually synchronized with Business Glossary when you want; however, you are notified if the two glossaries are out of sync, each time Data Architect is launched.

1.3.9 InfoSphere Information Server Manager, ISTOOL, and InfoSphere Metadata Asset Manager

Because each InfoSphere Information Server product module and component generates and consumes metadata to and from the InfoSphere Information Server repository, providing mechanisms for managing the repository is necessary. As such, InfoSphere Information Server has three main utilities for managing various aspects of the InfoSphere Information Server metadata repository:

- ▶ InfoSphere Information Server Manager
- ▶ ISTOOL
- ▶ InfoSphere Metadata Asset Manager

The *InfoSphere Information Server Manager Console* is a rich client user interface. It connects to one or more instances of InfoSphere Information Server and allows the administrator to organize DataStage and QualityStage objects (and optionally, their dependent objects) from one or more DataStage or QualityStage projects, into packages. These packages can be exported as files for version control (and subsequent deployment and import), or deployed directly into another instance of InfoSphere Information Server, such as from development to test, test to preproduction, or whatever other promotion scenarios fit the deployment architecture of the customer.

After packages are *defined* by using the InfoSphere Information Server Manager Console, the file creation (export) and deployment (import) of the package files can *also* be done by a command-line utility (CLI) named `istool`. The `istool` CLI is installed on both the client workstation and on the InfoSphere Information Server host. It can be initiated interactively by an administrator, or scripted for standardized use. A common practice is for the `istool` CLI file-creation and deployment scripts to be run by the enterprise scheduler for automating the process and maximizing security.

In addition to DataStage and QualityStage package-file creation and deployment functionality, the `istool` CLI is used for exporting the metadata created from all other InfoSphere Information Server product modules and components into `.ISX` archive files, such as the following items:

- ▶ Business Glossary: terms and categories
- ▶ FastTrack: projects and mapping specification objects
- ▶ Information Analyzer: projects and rules
- ▶ Shared (common) repository: physical data resources (common metadata)

These archive files can then be imported into other instances of InfoSphere Information Server, just like DataStage and QualityStage package files, whether

for development cycle deployment (DEV-TEST-PROD) or migration to a newer version of InfoSphere Information Server.

InfoSphere Metadata Asset Manager is a web-based component that provides several functions to the InfoSphere Information Server suite:

- ▶ Imports metadata from external sources (RDBMS, business intelligence, modelling tools, data files) into a staging area for comparisons with existing metadata for the purpose of manual conflict resolution.
- ▶ Loads approved metadata from staging area into the repository.
- ▶ Manages duplicate metadata in the repository (merge and delete).

InfoSphere Metadata Asset Manager uses Metadata Integration Bridges, which can translate metadata from external sources into formats that can be loaded into, and used by, InfoSphere Information Server. However, it also uses the same “connector” functionality that is used by DataStage and QualityStage, and Information Analyzer and FastTrack, for connecting directly to compatible RDBMS and ODBC data sources. Furthermore, it replaces most of the Import Export Manager functionality with an enhanced interface, that provides the additional functionality previously described.



InfoSphere Information Server installation topologies

This chapter describes installation topologies for IBM InfoSphere Information Server, ranging from basic single computer to highly available clusters and grid configurations.

This chapter considers deployment architectures for each of the major software subsystem components (tiers) of InfoSphere Information Server in combination and individually. The focus however is constrained to the perspective of a single overall installation environment, for example development, test, or production. Subsequent chapters describe multiple environment considerations. This chapter describes the following topologies and topics:

- ▶ Single computer
- ▶ Client server
- ▶ Dedicated computer per tier, and dedicated engine
- ▶ Clusters
- ▶ Active-passive and active-active
- ▶ Massively parallel
- ▶ Grid
- ▶ Web services
- ▶ Disaster recovery

2.1 InfoSphere Information Server software tiers

As Figure 2-1 on page 21 shows, InfoSphere Information Server has four tiers (one client tier and three server tiers):

- ▶ **Client tier:** This client tier presents the product interface to users through a Windows interface. Installed client components consist of Java and .Net components.
- ▶ **Services tier:** This server tier delivers common and product specific services that use WebSphere Application Server. The product software for Business Glossary and Metadata Workbench are principally hosted within the services tier and also components for the remaining product modules and common services across the product modules.
- ▶ **Engine tier:** This server tier delivers the parallel engine and framework for DataStage, QualityStage, and Information Analyzer. It also contains the database connectors, packs, service agents and DataStage server components.
- ▶ **InfoSphere Information Server repository tier:** This server tier provides a metadata persistence layer that is implemented by using a database. The database contains business, operational, and technical metadata including DataStage job designs and specifications.

Figure 2-1 also shows that 1 through N instances of the client, services, and engine tiers can be configured to use a single repository.

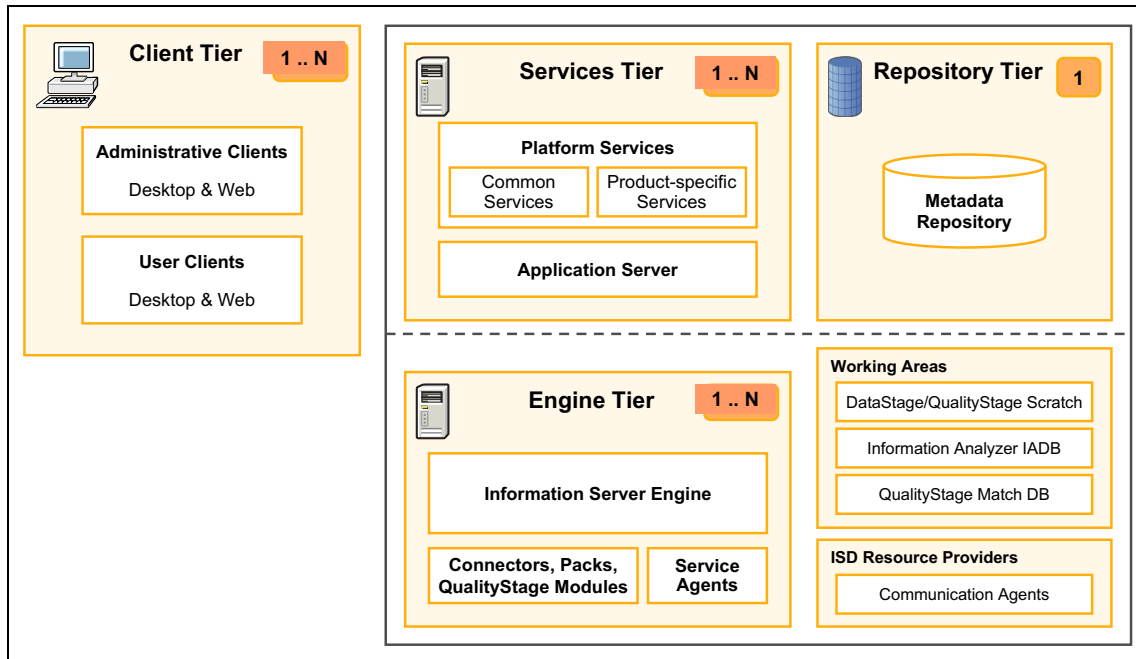


Figure 2-1 InfoSphere Information Server software tiers

2.2 Supported platforms

InfoSphere Information Server tiers are available on the following platforms:

- ▶ The installable client tier components, which provide the user interface, are available only on Windows platforms. Business Glossary and Metadata Workbench require only a supported web browser; there are no client installable components for these two InfoSphere Information Server modules.
- ▶ The server tiers (services, engine, repository) are available on UNIX, Linux, and Windows platforms (Microsoft Windows Server, Red Hat Linux, SUSE Linux, IBM AIX®, Oracle Sun Solaris, and Hewlett Packard HP-UX). Although each services tier component can be deployed on a separate host, the services and engine tiers should be deployed on the same platform type.
- ▶ The database for the InfoSphere Information Server repository may be implemented by using DB2, Oracle, or SQL Server.

In deployment topologies, where clients are connected to the server tiers over communication links that have poor bandwidth, latency, or both (for example, because of long distance), consider using a remote desktop (for example, terminal services or Citrix installations for the client access architecture).

The system requirements for InfoSphere Information Server vary, based on the scope and scale of the system. The exact configuration needed to support the environment with satisfactory performance can vary depending on multiple factors, such as server speed, memory, disk I/O, data volumes, and network and server workload, in addition to requirements regarding system or environment availability. (Capacity planning is out of scope of this book. However, you may request help from IBM technical professionals or IBM Techline when you size a specific implementation.)

The remainder of this chapter describes various installation topologies for a single environment installation, with benefits and considerations for each. Not every possible technical combination of tiers and clusters are described, however this chapter describes those combinations and principles that apply, so that you can make more informed choices for any given situation.

2.3 Single computer topology

In this configuration, all four tiers are on the same computer, as Figure 2-2 shows. This topology is suitable for demonstration systems and for small-scale development.

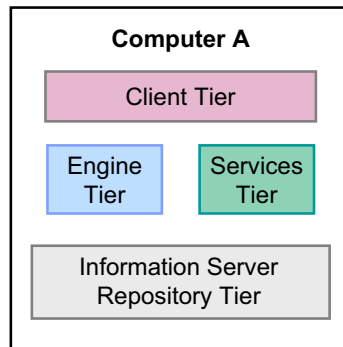


Figure 2-2 Single computer installation

Benefits and considerations of this topology

The benefits are indicated in the following list:

- ▶ Simple hardware deployment
- ▶ Simple to install
- ▶ No cross-server network or security configuration
- ▶ Suitably sized notebook installation possible

The considerations are indicated in the following list:

- ▶ Because of client requirements, this configuration inherently requires that all tiers be implemented on a supported Microsoft Windows platform. This consideration might be a concern for organizations that mainly use only UNIX or Linux server environments.
- ▶ Scalability is a consideration because all four tiers compete for the same machine resources.
- ▶ A hardware failure can likely result in total loss of service.
- ▶ Multiple users require terminal services.

2.4 Client server installation topology

In this topology, the services tier, engine tier, and InfoSphere Information Server repository tier are all installed on a single computer. The client tier is installed on separate computers.

The client tier computers that host installable client modules must run Microsoft Windows. The computer that hosts the other tiers can run any operating system that IBM InfoSphere Information Server supports.

This topology, shown in Figure 2-3, centralizes administration and isolates it from client users. Each user would have a client tier computer.

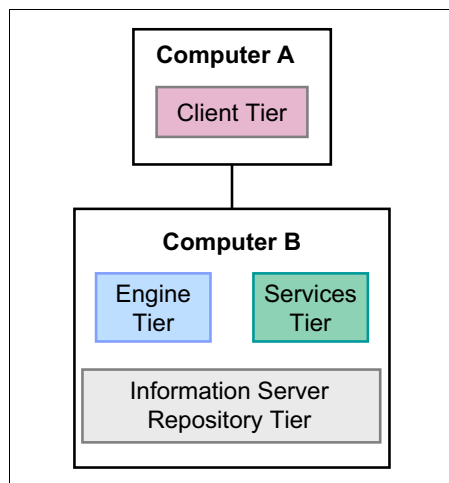


Figure 2-3 Client server topology

Benefits and considerations of this topology

The benefits are indicated in the following list:

- ▶ Full flexibility of choice of server operating environment hardware aligns with platforms that are supported for InfoSphere Information Server. You have full flexibility in the choice of a server operating environment when your hardware platform is one that is supported by InfoSphere Information Server.
- ▶ Clearer separation of administration of client and server components.

- ▶ High availability (HA) and failover are fairly simple and are server-based across a shared file system or a SAN.
- ▶ In specific comparison with the topology described in 2.3, “Single computer topology” on page 23, this topology also has the following benefits:
 - Concurrent clients are supported without needing terminal services for more than one concurrent local user.
 - A single client hardware failure does not affect the server or other clients.
 - Components that operate mainly within the client tier (for example Blueprint Director, InfoSphere Data Architect) remain available in the event of a server hardware outage.

Considerations are indicated in the following list:

- ▶ In non-trivial solutions, the server tiers compete for machine resources, particularly engine tier against other tiers.
- ▶ Failure of the server causes a complete outage of engine, services, and InfoSphere Information Server repository tiers.

2.5 Dedicated computer per tier topology

This topology is illustrated in Figure 2-4. Each tier has dedicated computing resources. Although the diagram shows only one client tier computer, this topology can include multiple client tier computers.

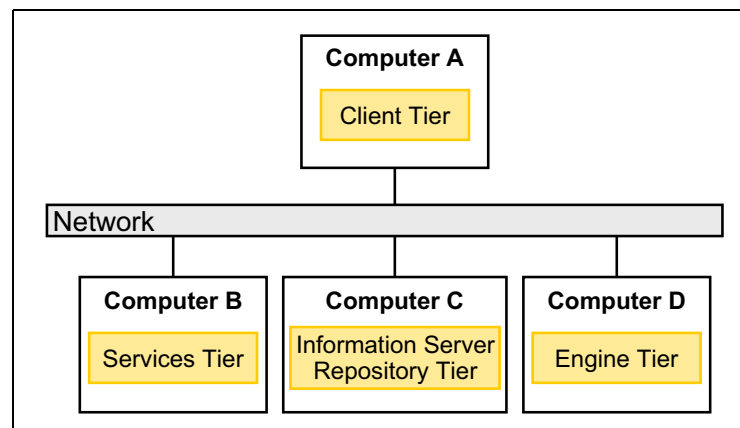


Figure 2-4 Dedicated computer per tier

Benefits and considerations of this topology

This topology has all the benefits of the topology in 2.4, “Client server installation topology” on page 24 plus the following benefits:

- ▶ Each server tier component may be sized and scaled separately according to load or evolving demands.
- ▶ Each server tier component may be managed separately. This might be a particular requirement of the repository (database) or services (WebSphere Application Server) tiers for some organizations.
- ▶ Each tier does not compete with another tier for the same machine resources according to load.
- ▶ In comparison with prior topologies, failure of the engine host machine does not cause failure of the services tier. Therefore in the loss of an engine tier server, some Business Glossary, Metadata Workbench, and FastTrack functions remain operational.

The considerations are indicated in the following list:

- ▶ Failure of the services or InfoSphere Information Server repository tier hosts causes a complete loss of access to all server-side functions for clients. For ways to extend this topology to mitigate this constraint, see 2.7, “InfoSphere Information Server clustered installations” on page 28.
- ▶ In topologies such as this one where the InfoSphere Information Server repository and services tiers are on separate hosts, a high bandwidth communication link must be maintained between those tiers, and those two particular tiers must be on the same subnet.
- ▶ Failover scenarios in this configuration are slightly more complex, because failover methodologies for each tier have to be setup individually.

2.6 Dedicated engine topology

In this topology, the services tier and InfoSphere Information Server repository tier are installed on one computer. The engine tier is installed on another computer. The client tier is installed on separate computers.

Figure 2-5 illustrates this topology. Although, for clarity, the diagram shows only one client tier computer, this topology typically includes multiple client tier computers.

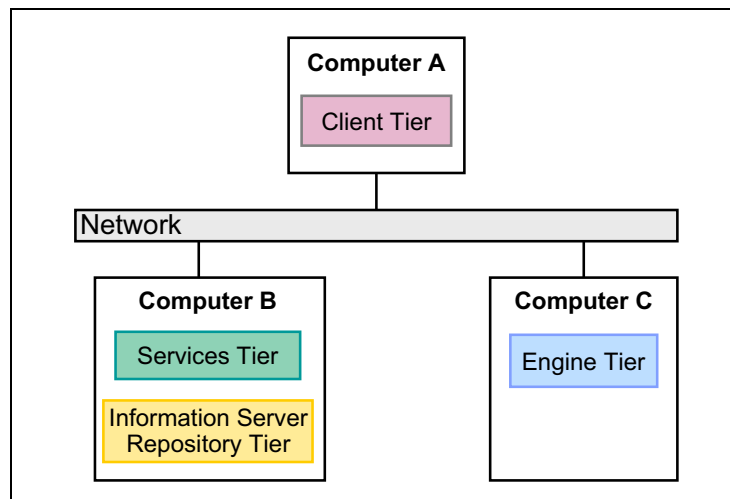


Figure 2-5 Dedicated engine

Benefits and considerations of this topology

The benefits are indicated in the following list:

- ▶ This topology has all the benefits of the prior topology described in 2.4, “Client server installation topology” on page 24
- ▶ Installing the InfoSphere Information Server repository tier with the services tier provides optimal performance because there is no network latency between the tiers. Also, higher engine tier activity does not affect the stages of the services tier and InfoSphere Information Server repository tier.
- ▶ Separating the engine tier is suitable for environments that have higher IBM InfoSphere DataStage and QualityStage job requirements, and for IBM InfoSphere Information Analyzer used in extensive profiling or data rules processing.
- ▶ Most Business Glossary, Metadata Workbench, and FastTrack functions remain operational in the case of an engine host machine failure.

One consideration of this topology is that the failure of the services or repository host causes a complete loss of service. For ways to extend this topology to mitigate this constraint, see 2.7, “InfoSphere Information Server clustered installations” on page 28.

2.7 InfoSphere Information Server clustered installations

Previous sections (2.3, “Single computer topology” on page 23 through 2.6, “Dedicated engine topology” on page 26) describe, at most, a single physical server per tier. This section describes clustering an InfoSphere Information Server installation.

Clustering an InfoSphere Information Server instance may be approached tier by tier, and also for an installation as a whole.

According to the choices made, clustering can bring benefits of higher availability or additional scalability.

Clustering InfoSphere Information Server also extends the range of deployment topologies into multiple engine installations, massively parallel processing (MPP) and grid configurations.

Two important terms are used in describing clustered installations:

- ▶ *High availability*: This term describes systems that are able to recover quickly when hardware failures occur.
- ▶ *Disaster recovery*: This term is the ability to recover a data center at a separate site, on separate hardware, if a disaster destroys the entire primary site or renders it inoperable.

Clustering the InfoSphere Information Server instance within the same data center can provide high availability within that data center, when correctly configured. Disaster recovery considerations for InfoSphere Information Server are described in 2.8, “Disaster recovery” on page 57.

2.7.1 Active-passive cluster

This section describes an active-passive cluster architecture that extends the topology described in 2.4, “Client server installation topology” on page 24. In the *active-passive* cluster topology, two server-side computers are involved, and the disk storage that contains InfoSphere Information Server binaries and data must be located on storage that is managed externally to those computers. The external storage must be able to be mounted or unmounted from one or the other server computer. This storage area is typically managed through a storage area network (SAN) device, or less commonly network attached storage (NAS).

In the topology described in this section, the InfoSphere Information Server repository tier, engine tier, and services tier are all installed on external storage.

The external storage is mounted on the active server. Therefore, at any one time, one of the computers (the active server) hosts the started InfoSphere Information Server instance. The other computer (the passive server) has the operating system started, but does not have the external storage with InfoSphere Information Server installation mounted or InfoSphere Information Server started.

Cluster software to provide high availability (HA) must be installed on the servers.

An example of required cluster software are as follows:

- ▶ IBM Tivoli® System Automation for Multiplatforms
- ▶ IBM PowerHA®
(formerly HACMP™ - High Availability Cluster Multiprocessing)
- ▶ Oracle Sun Cluster
- ▶ Hewlett-Packard Serviceguard
- ▶ Microsoft Cluster Services on Windows Servers

The HA cluster software maintains a *heartbeat*, which is a periodic signal that is sent from the active server to the passive server and that indicates that the active server is operational.

In the event of a hardware failure of the active server, the HA software unmounts the external storage from the previously active server, and mounts it on the passive server, and then InfoSphere Information Server is restarted there. The HA software also transparently routes new incoming client connections to the newly active host. Service continues this way on the now newly active server to allow for the administrators to handle the failed host. Following rectification of the failure, the previously failed host then becomes the passive host in the cluster, and the cluster continues from there.

The external storage system must also internally guarantee that any single disk failure is handled transparently, which is a standard feature of SAN and NAS storage systems. The ability of SAN or NAS systems to handle single-disk failures is typically achieved by using Redundant Array of Independent Disks (RAID) technology.

Figure 2-6 illustrates this topology. As before, although the diagram shows only one client tier computer for clarity, multiple client tier computers can be used.

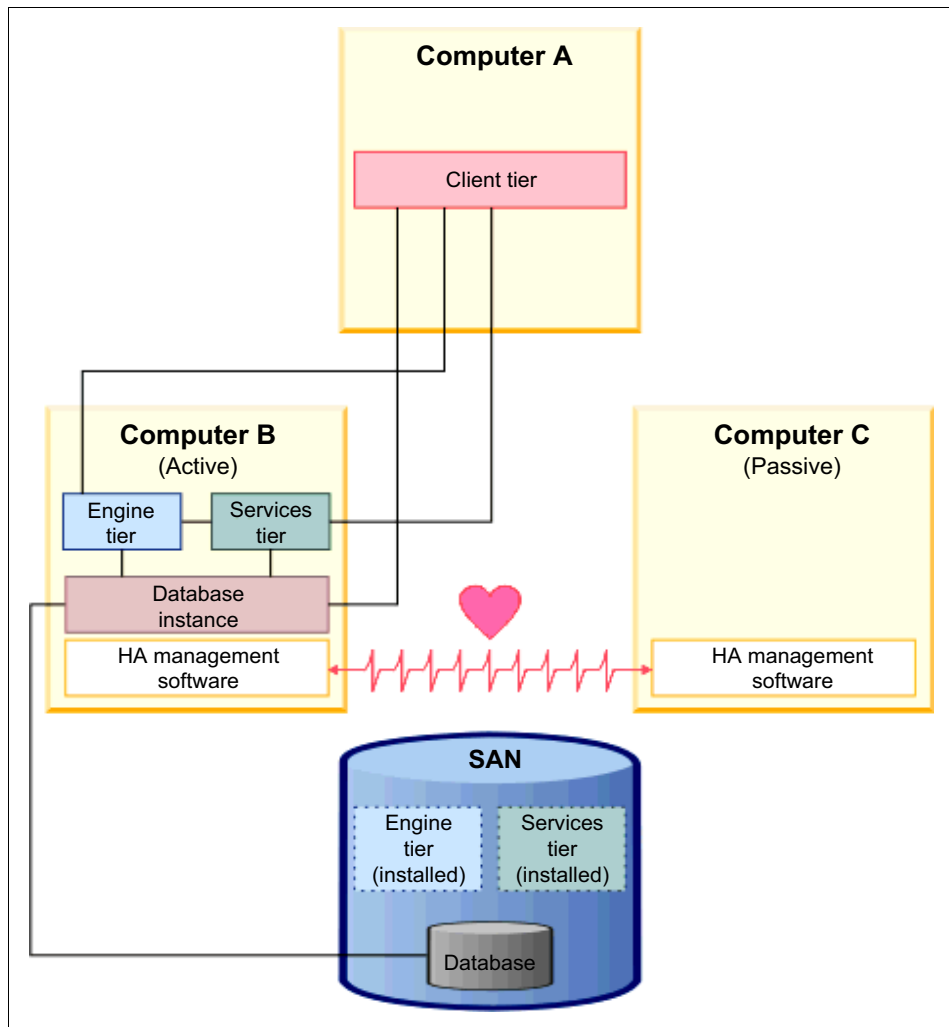


Figure 2-6 Active-passive topology

Figure 2-7 illustrates this topology in failover mode.

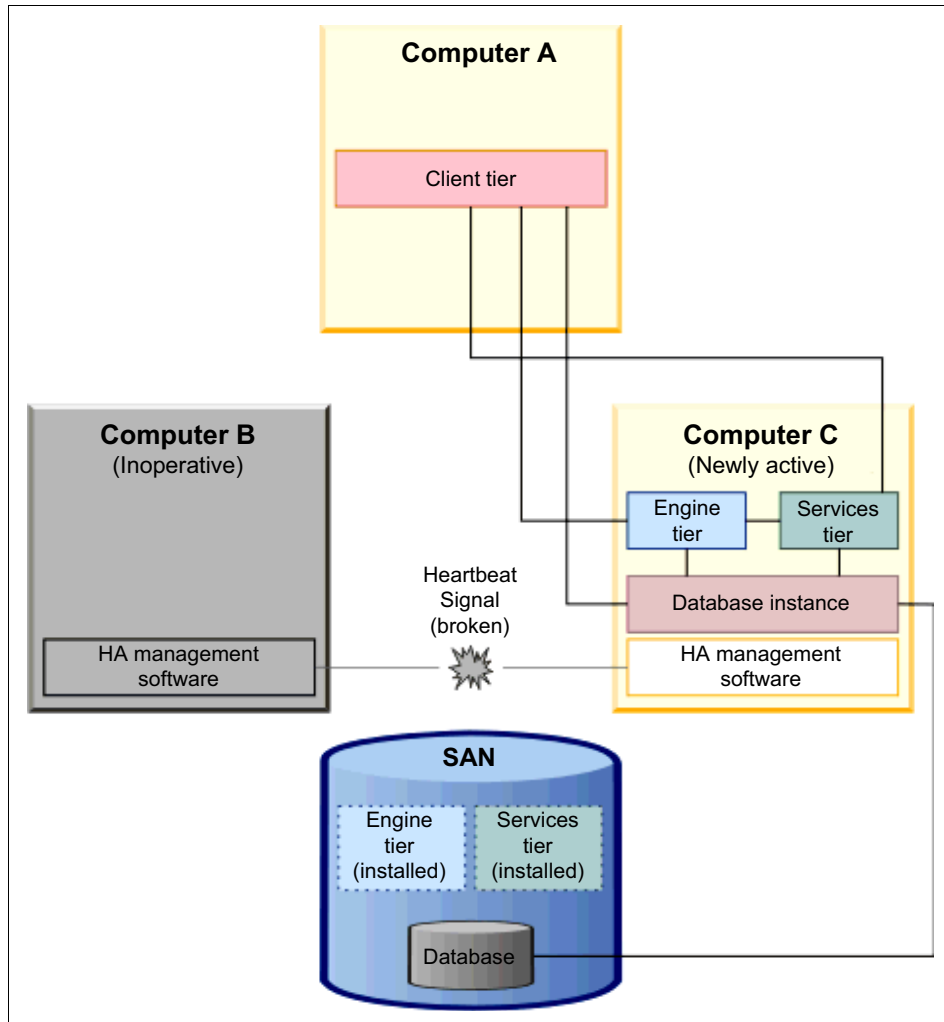


Figure 2-7 Active-passive cluster in failover mode

To enable the clients to automatically route to the currently active node, the HA management software must provide a virtual host name facility. Clients use the virtual host name as the point of connection to the cluster. The HA management software maintains this virtual host name irrespective of which active server computer is in use, and routes incoming connections to the currently active host through an internal virtual to physical mapping table.

Benefits and considerations of this topology

The benefits are indicated in the following list:

- ▶ In comparison with client server topology, this solution provides much higher availability because a hardware failure of the host for the server tiers does not cause an outage that lasts until rectification of the originating hardware fault.
- ▶ HA cluster software is selectable according to departmental or data center standards. The cluster software must at least provide customizable heartbeat, virtual host name functions and ability to script actions on a failover sequence.
- ▶ Additional options are available to reduce outage for system administrators who need to apply operating system patches or server firmware upgrades.

The considerations are indicated in the following list:

- ▶ Slight additional installation complexity is involved.
- ▶ The standby host is always completely idle. This constraint may be alleviated by using an active-active topology, described in 2.7.6, “Active-active engine tier topology” on page 47.
- ▶ The failover scripts might take a few minutes to start the application server on the failover node, so the outage period during failover matches this time.
- ▶ This topology can be deployed in development and also production installation environments. However, if failover occurs in a development environment, and, at the time of failover, there are open unsaved design time changes in the client, those changes must be re-keyed following the InfoSphere Information Server restart operation on the failover node.
- ▶ There is no improvement on scalability over basic client server topology.

The following sections in this chapter elaborate further clustering options at a tier-by-tier level which can further improve upon these constraints. More complex arrangements are possible when you consider each tier separately, but consequent benefits are also possible. The benefits and considerations are described in the sections that follow.

2.7.2 Clustering the services tier

Figure 2-8 on page 33, shows a summary of the broad range of InfoSphere Information Server functions that are hosted within the services tier on WebSphere Application Server. For a single host that is running WebSphere Application Server, a failure in this layer can cause a complete InfoSphere Information Server outage because of the loss of these backbone services. Failures might occur because of hardware failure of the host machine. Similarly, overall service might be degraded or users can experience poor response times

if the processing load of these services exhausts the machine resources that are available.

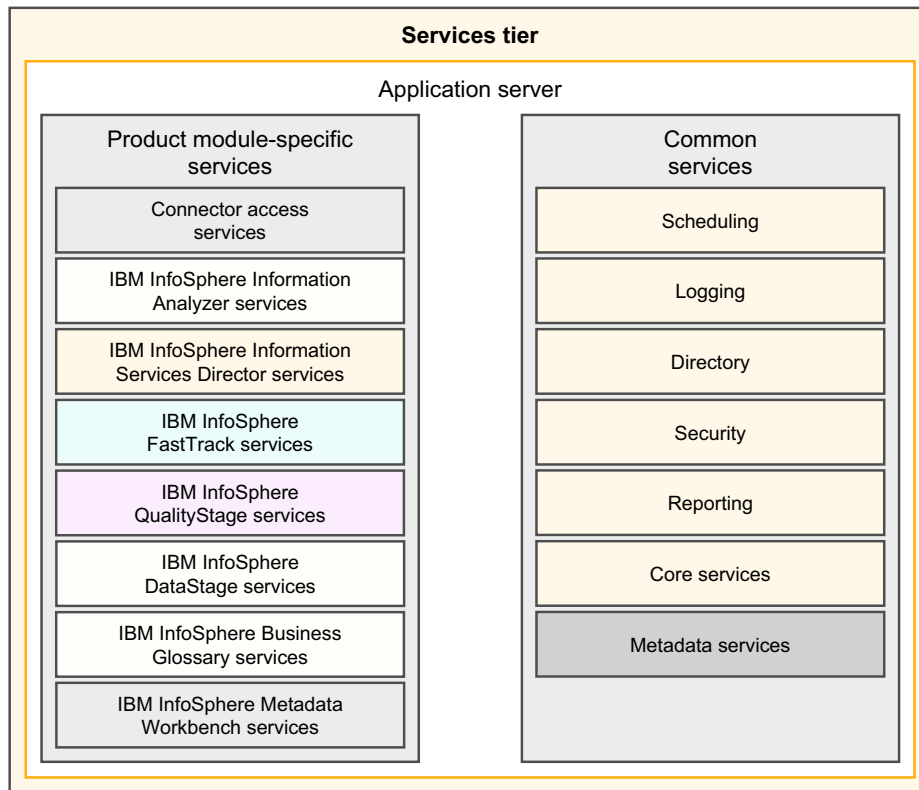


Figure 2-8 Functions hosted in WebSphere Application Server

From InfoSphere Information Server version 8.5 and later, the product installation bundles the WebSphere Application Server Network Deployment edition. This version of the application server has features that can be used to support both load balancing and high availability architectures around the services tier.

WebSphere Application Server Network Deployment supports active-active clusters and also an array of N redundant nodes across which the overall installation workload is managed. Additional nodes can be added to give full scalability of the services tier and so those aspects of InfoSphere Information Server functionality that are hosted within WebSphere Application Server can grow and scale according to the requirement for concurrent users and resource demands.

To make use of this feature, a front-end web server with WebSphere Application Server Workload Manager (WLM) plug-in is deployed, or a load balancer, or both. These front-end components are referred to as the *front-end dispatcher*.

Using a web server front-end

Figure 2-9 shows a front-end web server with the WLM plug-in. The supported web servers are either IBM HTTP Web Server or Apache Web Server (Apache), from the Apache Software Foundation. To avoid a single point of failure, use a backup web server to fully and quickly take over in the event that the primary web server fails.

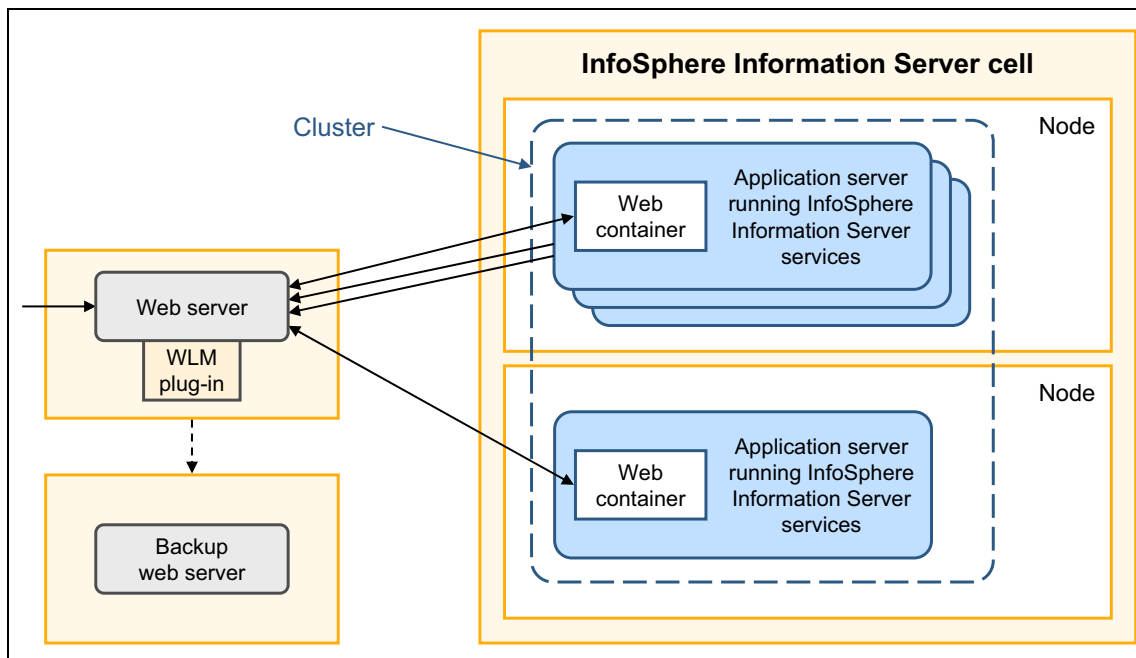


Figure 2-9 Services tier load-balancing using web services

In a development environment the web server can be located on the same servers as WebSphere Application Server. In a production environment, security considerations might require a firewall between the web servers and WebSphere Application Server. In that case, they can be installed on separate servers.

In this type of clustered WebSphere Application Server installation, as clients connect and create new sessions within WebSphere Application Server, the session is allocated to the node according to the workload management algorithm that is defined. After a session is created within a particular server, the session runs to completion within that node. WebSphere Application Server

replicates the session to all WebSphere Application Server cluster members, so that in the event of a cluster member failure, the session can be resumed on another cluster member with the minimum of outage time (typically seconds).

Using a load balancer front end

As an alternative to using web servers, load balancers can be used. Figure 2-10 illustrates a services tier topology by using this approach.

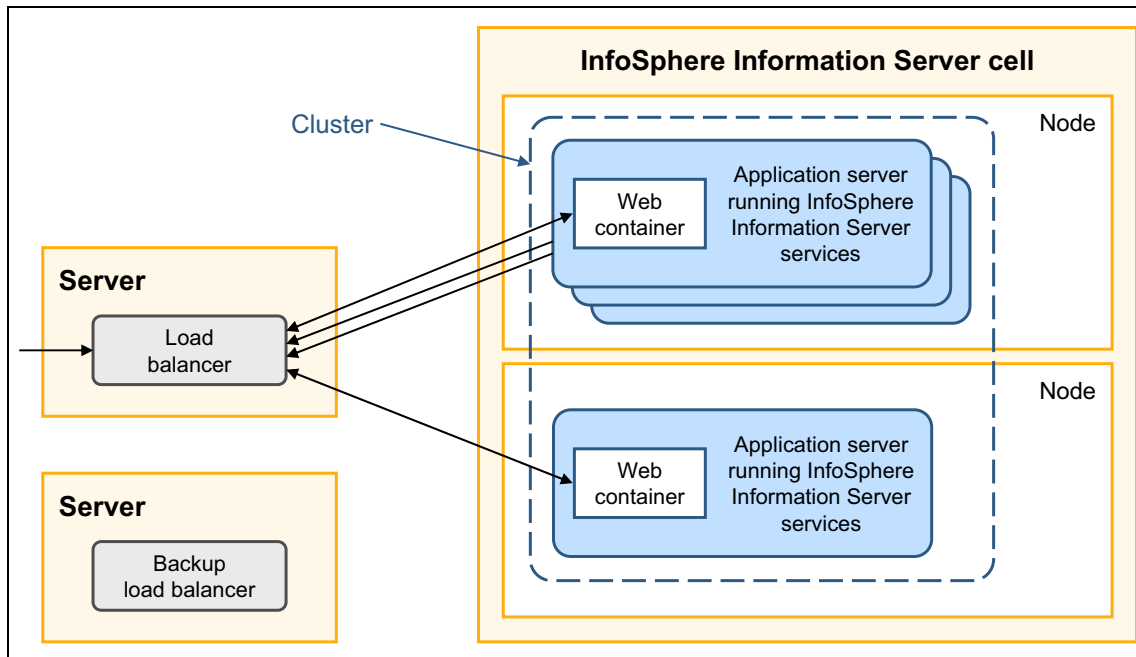


Figure 2-10 Services tier using explicit load balancers

Load balancers can be implemented by using any hardware or software load balancer; the only requirement is that the choice of load balancer must declare support for WebSphere Application Server.

- ▶ An example of a hardware load balancer is a product family named BIG-IP from F5 Networks, Inc.
- ▶ An example of a software load balancer is IBM WebSphere Edge Components for WebSphere Application Server.

The choice of load balancer might be driven by the data center or architecture standards of an organization. However, in addition, load balancers can offer more complex workload management when compared to the WLM plug-in for the web server previously described.

Load balancers can be more complex to set up by comparison; for example, session affinity is more complex to set up in a load balancer and might need the involvement of a network specialist. Hardware load balancers can also be more costly.

A preferred approach is for the session affinity timeout in the load balancer to be maintained to the same value that is set as the InfoSphere Information Server session timeout in WebSphere Application Server. By default this value is 30 minutes. This value might be increased in a DataStage development environment to match the value that is set in DataStage Administrator for the timeout to release DataStage resources.

Following a period of inactivity given by the affinity timeout value, the load balancers will assign or reassign any subsequent request from the client by using the load balancing algorithm again.

Using a load balancer and web server front end

The front end might also consist of a combination of a load balancer and a web server. To obtain high availability and the maximum balancing of server and workload capacity, deploy a load balancer upstream of a group of web servers. This way is referred to as an *IP sprayer* architecture.

The load balancer performs intelligent balancing among the web servers, based on server availability and workload capacity. Choose this topology to eliminate a single point of failure at the web server level. This arrangement also spreads the load of web requests across multiple web servers. InfoSphere Information Server supports many IP sprayer solutions that are manufactured by IBM and other vendors.

To prevent single points of failure at the load balancer level, deploy a backup load balancer to take over in case the active one fails.

Figure 2-11 shows an IP sprayer topology. The diagram shows an InfoSphere Information Server cluster with two front-end web servers. A load balancer is deployed upstream of the web servers. A backup load balancer is also deployed. The web servers and load balancers are installed on separate servers.

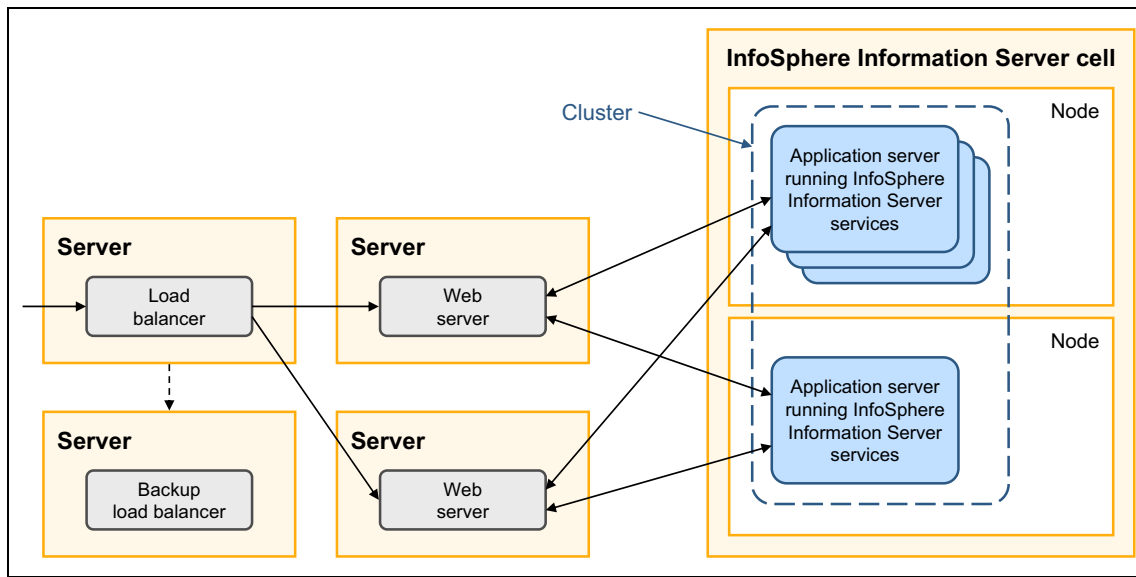


Figure 2-11 IP sprayer front end

Benefits and considerations of clustering the services tier

A description of clustering all server tiers together is provided in 2.7.1, “Active-passive cluster” on page 28. Although that option delivers high availability, an outage period during the failover process might be as much as 10 - 15 minutes to fully restart. That option also requires an idle server 100% of the time.

Considering the services tier alone, there are significant benefits of clustering the services tier in scalability, load balancing, and availability of the InfoSphere Information Server functions that are hosted in this tier in comparison to the whole-suite approach. The time to fail over a services tier will be likely a few seconds; any servers that are involved in the services tier infrastructure are available to do work during normal operation on behalf of the overall InfoSphere Information Server installation.

These benefits are gained with some additional complexity of initial setup and configuration of the associated servers and the requirement to add front-end dispatcher functionality, delivered by using either web servers, load balancers (hardware or software), or both.

Licensing calculation: In calculating licensing, all active processor value units (PVUs) are considered here. (By comparison, in an active-passive cluster, the PVUs are calculated only for the active server.)

2.7.3 Clustering the InfoSphere Information Server repository tier

Creating a database cluster for the InfoSphere Information Server repository tier is possible.

InfoSphere Information Server supports the InfoSphere Information Server repository tier, implemented by using IBM DB2, Oracle Corporation Oracle Database, or Microsoft SQL Server. The InfoSphere Information Server installer includes a license of IBM DB2 that is freely entitled for use as the InfoSphere Information Server repository database. The choice of InfoSphere Information Server repository tier database technology that is used can also be driven by customer-preferred database technology and availability of local database administration skills.

The precise details of the clustering implementation differ according to the database technology adopted in the InfoSphere Information Server repository tier. At a higher level, however, details are all similar: two or more physical servers are required to host the database for redundancy. Cluster capabilities of the database handle a server outage with the net effect that the InfoSphere Information Server connection is directed to remaining database servers.

The remainder of this section describes details and options that are available for clustering the InfoSphere Information Server repository tier according to the selected database technology: DB2, Oracle Database, or SQL Server.

Clustering a DB2 InfoSphere Information Server repository tier

InfoSphere Information Server supports the following choices for clustering the DB2 InfoSphere Information Server repository tier:

- ▶ Active-passive cluster
- ▶ HADR cluster

Active-passive cluster

In this configuration, the InfoSphere Information Server repository database is shared between nodes in the cluster. If a fail over occurs, another node in the

cluster provides DB2 functionality. To provide high availability, set up your cluster in an active-passive configuration with a single active DB2 instance on one computer, and one or more passive instances on the other computers. If the DB2 instance encounters a problem or fails, a passive instance can take over.

To manage this configuration, DB2 includes, as standard, all the components of Tivoli System Automation cluster software. This functionality is built in to DB2 since version 9.5 and is included in the InfoSphere Information Server license.

Tivoli System Automation software maintains a heartbeat signal between the nodes in the cluster. If the heartbeat fails on the active node, the software initiates failover to the passive node. This configuration is similar to that described in 2.7.1, “Active-passive cluster” on page 28; the primary difference is that the cluster is only around the DB2 hosted InfoSphere Information Server repository tier.

With this configuration, the DB2 failover is automatic, but might take several minutes as the new instance acquires resources, repeats certain transactions, and undoes other transactions. To minimize interruption and manual intervention, configure DB2 automatic client reroute (ACR). This function causes other components in the InfoSphere Information Server instance, such as WebSphere Application Server JDBC connection, to automatically reconnect to the new DB2 instance.

This configuration does not provide disaster recovery for the database itself. Instead, it provides high availability for database client processes and smooths the reconnection to the new node. To provide redundancy for the database itself, implement the high availability disaster recovery (HADR) feature of DB2.

HADR cluster

The DB2 high availability disaster recovery (HADR) feature is a database log replication feature that can be used as part of a high availability and disaster recovery design for the InfoSphere Information Server repository tier, including for both partial and complete site failures. HADR protects against data loss by continually replicating data changes from a source (primary) database, to a target (standby) database. Clients that were using the original primary database can be directed to the standby database (which becomes the new primary database) by using automatic client reroute (ACR) and reconnect logic in the application.

DB2 HADR works in a similar fashion to the log-shipping solutions, but all the work is done inside DB2 at the software level. A DB2 HADR primary database uses internal processes to ship database log buffers to an HADR standby database. A process at the standby server then replays the log records directly to the standby database. The standby database can be converted to the primary

database and accessed by InfoSphere Information Server in the event of failover when the primary fails.

Use HADR in tandem with the cluster software features of Tivoli System Automation. During a primary database failure, Tivoli System Automation clustering software adds automated HADR takeover. Tivoli adds the monitoring of the primary database server for outages and then monitors the environment of the primary database server for outages. As illustrated in Figure 2-12, the standby database keeps waiting for log records to be transferred although the primary database is no longer active. Without cluster software, monitoring the HADR pair and manually issuing appropriate takeover commands is necessary in the event of a primary database server failure. During a primary database server failure, clustering software is required to automate HADR takeover.

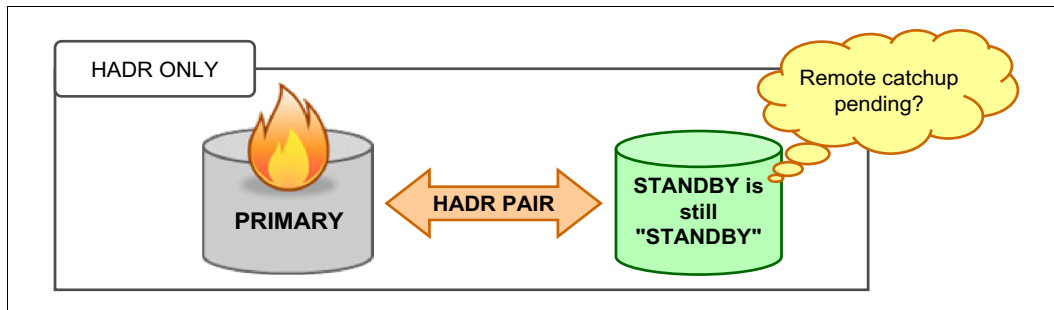


Figure 2-12 HADR standby remains waiting

The bundled Tivoli System Automation cluster software can be set up functionally to issue the takeover commands to the standby HADR node. This node can then automatically become the primary, as Figure 2-13 shows.

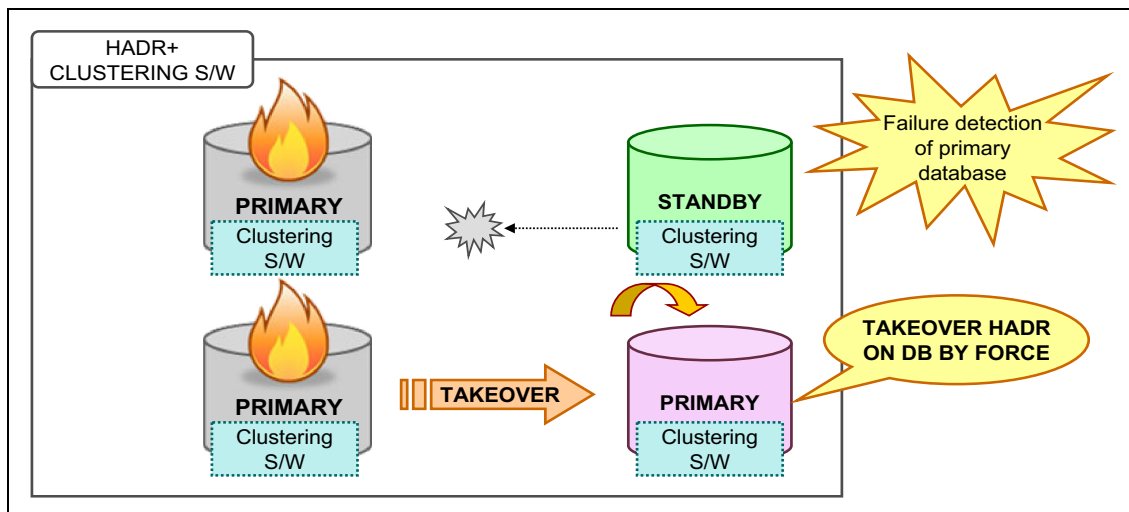


Figure 2-13 Cluster software issues HADR takeover command

Benefits and considerations of this topology

The benefits are indicated in the following list:

- ▶ Capability exists for fast failover (typically under one minute).
- ▶ Performance has a negligible impact.
- ▶ Rolling upgrades without service interruption between non-major versions or changes requiring server recycling, and reduced change windows for other upgrades.
- ▶ The standby database can be used in read-only mode; read-only mode can be useful for administration or support.
- ▶ The standby database can be at a remote data center (see 2.8, “Disaster recovery” on page 57).

A consideration is that some additional complexity is required for setup in comparison with an active-passive cluster.

Clustering an Oracle Information Server repository tier

Oracle Corporation provides software for clustering and high availability in the Oracle database. The options available are Oracle Real Application Clusters and Oracle Data Guard.

Oracle Real Application Clusters (RAC)

Oracle RAC allows multiple computers to run the Oracle Relational Database Management System at the same time, while providing access to a single database. External clients may connect to any of the servers involved in the RAC and have access to a coordinated set of data.

Because all computers involved in the database cluster must present the same data to clients, the overall system must guarantee that database data changes are visible across the cluster so that all clients can see the current version of data. Oracle refers to the RAC functionality that delivers this capability as *Cache Fusion*. To achieve this capability, Oracle RAC clusters require a fast dedicated internal network between the RAC servers involved.

Oracle advises that if a server in the cluster fails, the Oracle database continues to run in the remaining servers in the cluster. Also individual servers can be shut down for maintenance while database clients can continue to operate.

From Oracle Database Version 11g Release 2 (11gR2, which is a version supported by InfoSphere Information Server at time of writing), workload management is also included. In this case, connections are load-balanced across the server pool and performance is tracked on a per-service basis by the Oracle Database 11g Automatic Workload Repository facility.

See the Oracle documentation for more information about Oracle Real Application Clusters.

Oracle Data Guard and Oracle Active Data Guard

The software that Oracle Corporation markets as Oracle Data Guard forms an extension to the Oracle RDBMS and is available in the Enterprise Edition. It aids in establishing and maintaining databases in the roles of “secondary standby” and as alternative repositories in the role of the “primary” databases. Databases have the ability to transition from one role to the other.

Data Guard supports both physical standby and logical standby sites:

- ▶ A physical standby database replicates the exact contents of its primary database across the Oracle Net network layer.
- ▶ A logical standby database converts the redo that is generated at the primary database into data and SQL and then reapplies those SQL transactions on the logical standby.

Oracle Active Data Guard extends Oracle Data Guard physical standby functionality in Oracle 11g configurations to allow read-only access on the standby node at the same time as archiving transactions from the primary node.

Clustering a Microsoft SQL Server Information Server repository tier

For customers who use SQL Server as their preferred database technology, InfoSphere Information Server also supports an InfoSphere Information Server repository tier on SQL Server. At time of writing, InfoSphere Information Server 8.7 supports SQL Server Enterprise Edition 2005 SP2 or 2008.

For clustering Windows servers, Microsoft has a technology named Windows Server Failover Clustering, which was previously known as Microsoft Cluster Service (MSCS).

An individual InfoSphere Information Server repository can be set up in SQL Server, where the SQL Server database is hosted in an active-passive cluster that uses Microsoft Cluster Service or Windows Server Failover Clustering. For the description of an active-passive topology cluster basics, see 2.7.1, “Active-passive cluster” on page 28.

See the Microsoft documentation for more information about Microsoft SQL Server clusters.

2.7.4 Engine tier clustering

The InfoSphere Information Server parallel engine has a powerful and flexible architecture that supports a variety of cluster types, from a single server to hundreds of servers.

To more fully appreciate the options for clustering the engine tier, a helpful approach is to understand, at a high level, the relevant parallel engine features:

- ▶ Pipeline parallelism
- ▶ Partition parallelism
- ▶ Parallel job configuration file

The parallel engine supports both pipeline and partition parallelism. The parallelism operates within each job and is the default behavior of jobs.

Pipeline and partition parallelism

A job consists of a number of stages. Examples of stages are lookup, join, and aggregate.

Pipeline parallelism is a feature of the engine in which stages in a job are able to process data in a pipeline. This feature allows downstream stages in the job to process the output of upstream stages, while the upstream stages move onto the next row, so that all stages are kept working at the same time.

Partition parallelism is used to describe the feature of the engine that may be conceptually considered as where several instances of each job stage are created at the time the job starts. As the job runs, the data is split into sets and each stage instance processes an allocated set of the data. Each stage instance processes its set of data during, and therefore in parallel with, the time when other instances are processing their allocated set. A set of data is referred to as a *partition*. The process of splitting the data into sets is referred to as *partitioning the data*, and the corresponding number of stage instances is referred to as the *degree of partitioning*.

Automatically performing both pipeline and partition parallelism enables data to be processed many times faster. Figure 2-14 illustrates a conceptual job that consists of three stages: transform, enrich, and load.

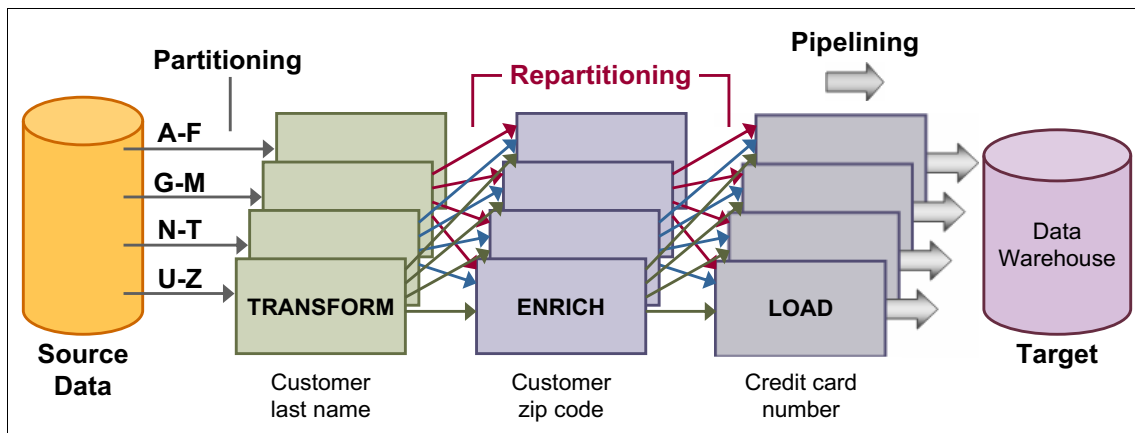


Figure 2-14 Partition and pipeline parallelism

All three stages run in a pipeline, so downstream rows are being loaded to the target warehouse at the same time as the upstream transform and enrichment stages are processing later rows that are extracted from the source. The job in the same figure also illustrates a four-way degree of partitioning: four instances of each operator are shown.

In this example, the transform operator is shown partitioning the data by the first letter of the customer's last name. Therefore one instance of the transform stage is processing customers with last names that begin A through F. At the same time, the second instance processes customers with surnames beginning G through M and so on.

By combining pipeline and partition parallelism, this conceptual job might process data at least 20 times faster than otherwise might occur.

Parallel engine configuration file

Parallel engine jobs support a runtime-assigned *configuration file*. There is at most one configuration file per job.

The configuration file describes the number of partitions that each job should run with. Each partition is described in the configuration file by reference to the server it runs on and disk and other resources. Each partition is referred to in the configuration file as a node, so in Figure 2-15 on page 46, a configuration file with four nodes was used.

The configuration file describes, for each node, what areas of the disk should be used to hold working data and temporary (scratch) areas, and also which computer host that the node should use. Therefore, all partitions of a job may run on a single server, or each node may be run on a separate server and use the same or different areas of disk, or any permutation.

This powerful feature means that jobs that need InfoSphere Information Server engine resources can easily be scaled up or down with workload by changing only the runtime parallel configuration file that is associated with that job; no other changes are required and no job recompilation is required. This feature also means that jobs can be run on a separate server or use a different number of servers in a cluster with only a change of the configuration file.

Figure 2-15 on page 46 shows a conceptual DataStage job that transforms information from tables in a source system, aggregates, and then loads the resulting data to tables in a data warehouse.

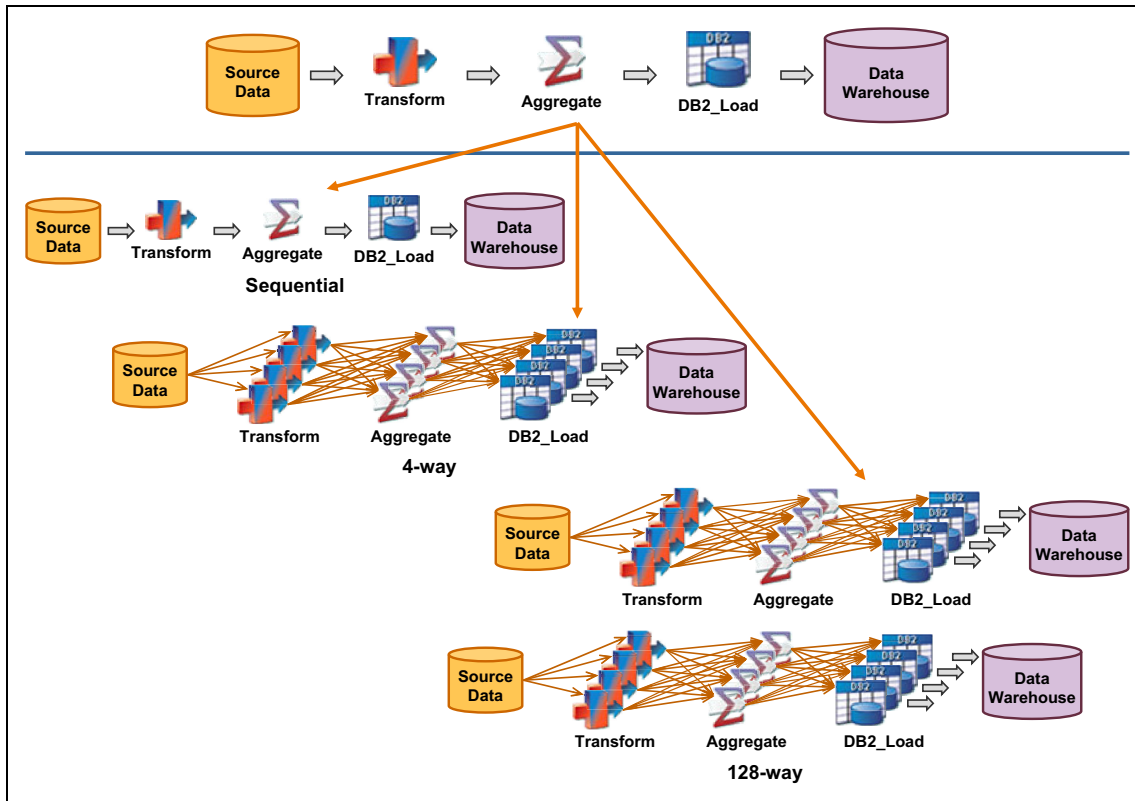


Figure 2-15 Run time selectable parallelism

The figure illustrates that the same job may be run with various degrees of partitioning by selecting the relevant configuration file, in this case sequentially (one node), partitioned 4 ways (4 nodes) or partitioned 128 ways (128 nodes).

Pipeline parallelism occurs irrespective of the number of nodes in the configuration file that is used by the job.

Based on a project's requirements, an integration solution may consist of tens or even hundreds of such jobs. In addition, customers often deploy several integration solutions on a single platform, so the aggregate number of jobs across one or more projects might reach thousands. The limits are determined by hardware platform capacity.

By having the flexibility of being able to assign nodes to jobs and also the flexibility to specify the degree of parallelism on a job-by-job basis, using the configuration file provides options for solution workload management, scalability, and cluster design that would otherwise not be possible.

2.7.5 Active-passive engine tier topology

This topology is similar to the option described in 2.7.1, “Active-passive cluster” on page 28; the difference is that in this option the engine tier is installed in its own active-passive cluster, separate from the other tiers. The other tiers might be separately clustered as described in prior sections.

As before, the cluster software provides a virtual host name pointing to the currently active host. In this option, the parallel engine configuration file supports partition parallelism on the currently active server and the node name is set to the virtual host name.

Benefits and considerations of this topology

The benefit is the high availability of the engine tier.

The considerations are indicated in the following list:

- ▶ Whichever side is the currently passive node is not used for engine workload.
- ▶ Jobs that were running at the time of the server failure must be restarted: A typical integration solution consisting of many jobs, which, when end to end in sequence, might run in normal operator for an hour or more. Solutions should use the checkpoint restart feature of the DataStage sequencer to enable restarting from the point of failure to minimize the impact of the outage during failover.

2.7.6 Active-active engine tier topology

This section describes the extensions to the active passive architecture introduced in 2.7.1, “Active-passive cluster” on page 28 and 2.7.5, “Active-passive engine tier topology” on page 47.

This topology is similar to active-passive in setup, with the primary difference being that the parallel engine configuration file can be used to run engine jobs on both engine nodes in the cluster.

In contrast to active-passive, in this topology, the engine tier software and application data areas must be simultaneously mounted on both servers in the engine tier cluster. As such, both cluster software and file-sharing software, such as cluster file system or IBM GPFS™ software, must be installed.

Where multiple engines are involved there is an installation benefit: there is still the requirement to perform a single installation of the engine. That single engine installation may be done through a node in the cluster because the installation image can be shared (the virtual host name should be provided).

With this topology, only a single installation of the engine is necessary, and that installation may be done through either node of the cluster, and the virtual host name must be provided.

Therefore all the InfoSphere Information Server engine software binaries, application data mount points, and disk paths are identical on both servers, and the cluster software provides a virtual host name to which clients and other tiers are connected.

In contrast to active-passive, the parallel engine configuration file contains the virtual host name for the conductor, but the physical host name (or host names) that the stages will run on for the other nodes. The *conductor* is the server that the parallel job is launched from.

Also in contrast to active-passive, during failover processing, the disk mount points are left unchanged because they are already mounted on both nodes. However, the cluster failover scripts must include changing all parallel engine configuration files so that only the remaining physical host is present.

2.7.7 Massively parallel processing (MPP) topology

Within an InfoSphere Information Server parallel engine massively parallel processing (MPP) topology, each processing node has its own CPUs, disks, and memory. In processing engine workload, these hardware resources are not shared across the nodes. This way is often referred to as a *shared nothing* architecture.

Figure 2-16 illustrates the comparative differences between shared nothing and symmetric multiprocessor (SMP) and uniprocessor architectures. Grid architecture, also in the figure, is refinement of MPP including dynamic workload management. Grid is described in more detail in 2.7.8, “Parallel Engine Grid topology” on page 52.

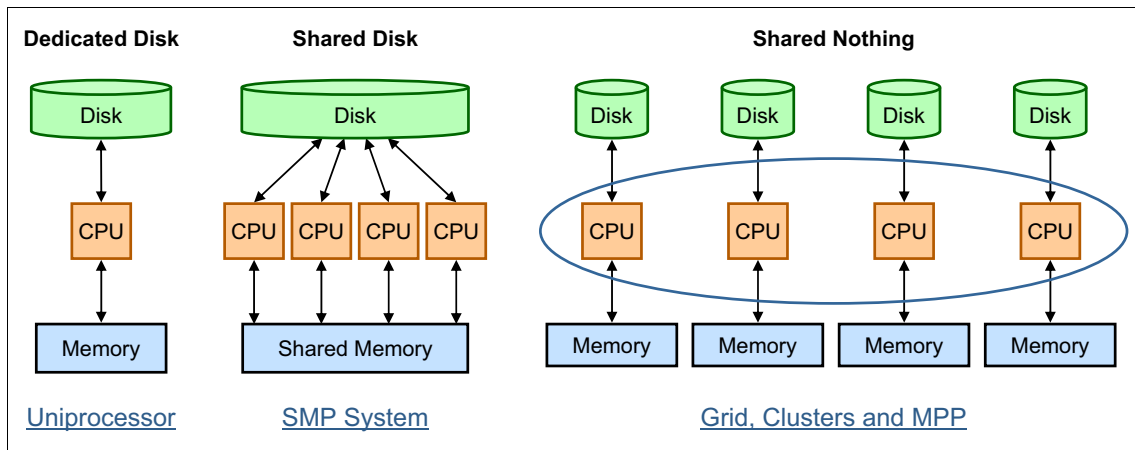


Figure 2-16 Computing architectures

Within InfoSphere Information Server parallel engine MPP architecture, linking the nodes by using a fast private network is advisable. The parallel engine job configuration file (that is introduced in “Parallel engine configuration file” on page 45) describes the network name of the node on the fast network that uses the fastname property.

Figure 2-17 shows a four-node MPP with a separate high-speed network for interconnect. In the figure, the _caa extension indicates the high speed network interfaces Domain Name System (DNS) names.

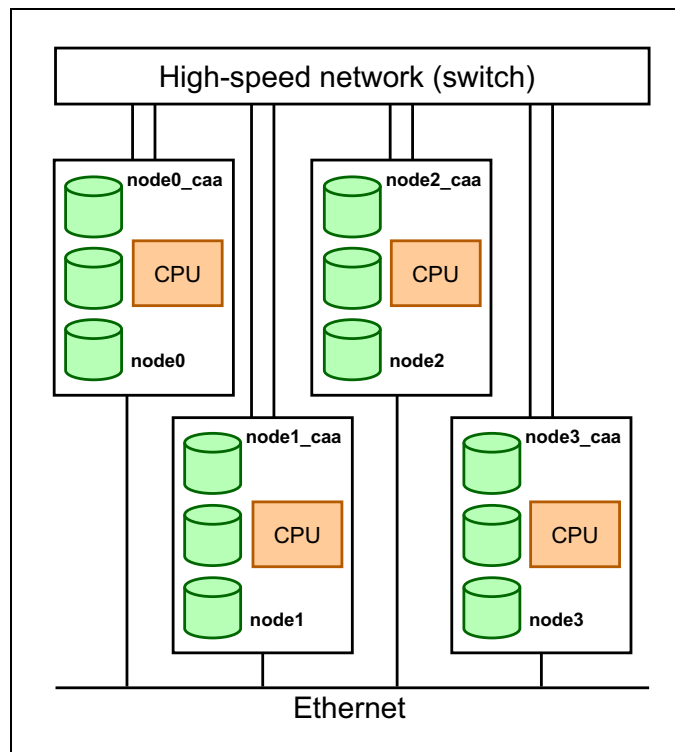


Figure 2-17 MPP with private network

Secure Shell (SSH) connections are typically used for cross-node authentication, but remote shell (rsh) can also be used if required.

Other tiers are omitted from the diagram for clarity.

Running within MPP, the resources that the job partition needs must all be accessible entirely from within that node.

In the design of an MPP solution, consider two types of processing nodes:

- ▶ Conductor node, also referred to as the head node
- ▶ Processing node, also referred to as compute node

When a parallel engine job is run, a hierarchy of system processes is created, as illustrated in Figure 2-18.

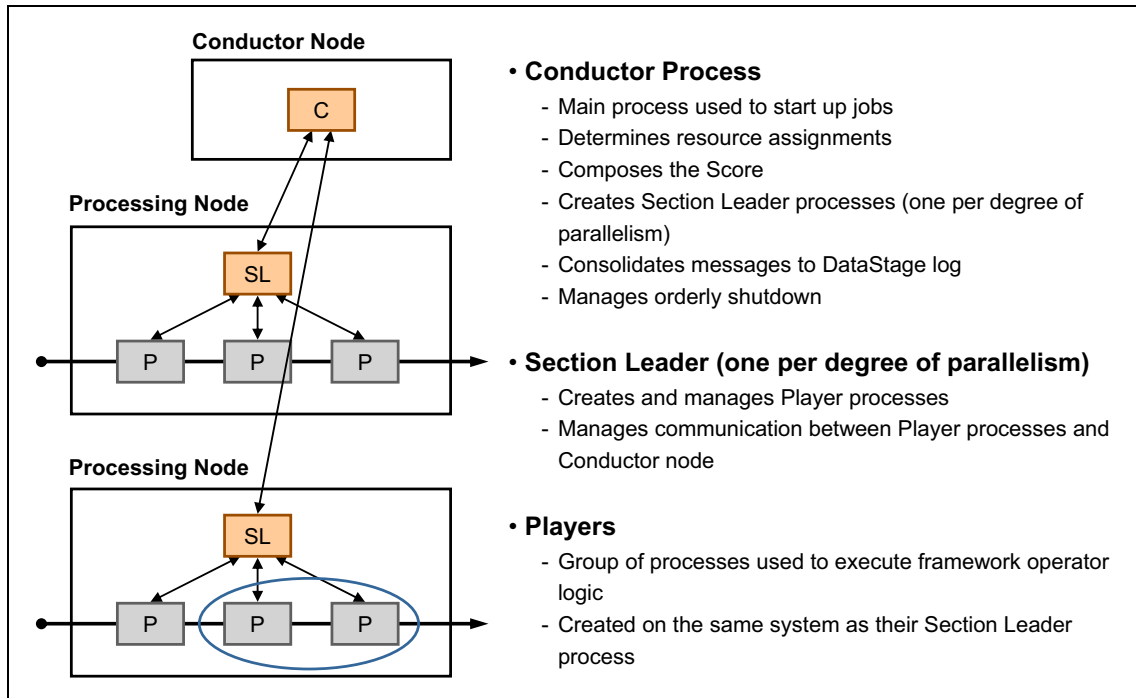


Figure 2-18 Parallel engine process hierarchy

The conductor process is responsible for the orderly creation and shutdown of the job, and the consolidation of job log messages from the collection of process that make up that job. The player processes are the stages that do the work on the job, and these processes therefore drive the majority of the server load. The process tree is designed to run both within and across servers that correspond to the information that is provided in the job configuration file.

The parallel engine conductor or head nodes would be lightly loaded if this is all that is running there. In MPP and grid architectures, the head or conductor node can be sized to also run the InfoSphere Information Server services tier, repository tier, or both tiers. This match is good for InfoSphere Information Server tier requirements because it ensures that the services and repository tiers are closely coupled, and the parallel engine processing nodes do not compete for server resources.

Figure 2-19 shows the head and compute nodes.

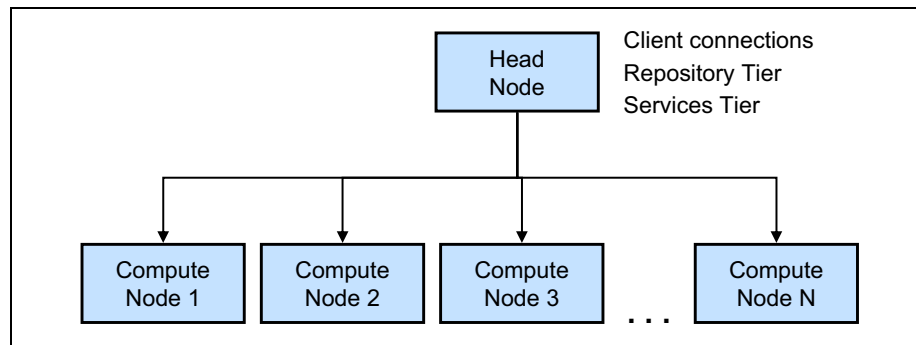


Figure 2-19 Head and compute nodes

Benefits and considerations of MPP engine tier topology

The benefits are indicated in the following list:

- ▶ All engine nodes are available to do data integration work while available.
- ▶ A failure of single server will have a corresponding fractional impact on the total server resources that are available.

The considerations are indicated in the following list:

- ▶ A private network is advised.
- ▶ InfoSphere Information Server engine binaries can either be copied to each node or be shared by using a shared file system that is mounted on all the nodes.
- ▶ The selection of the parallel configuration file that is used for each DataStage job, or sub sequence of a number of jobs, must be carefully selected; an important task is to consider which nodes that jobs *normally* run on when all servers are available, allowing for solution and other workload. As the number of jobs and projects increases, the selection and management of configuration files mapped to jobs, job run schedules, and projects can become increasingly complex to manage and maintain. Mitigating this issue is described in 2.7.8, “Parallel Engine Grid topology” on page 52.

2.7.8 Parallel Engine Grid topology

The grid engine topology can be considered an extension of the MPP topology, described in 2.7.7, “Massively parallel processing (MPP) topology” on page 49. This section describes the differences.

The grid topology adds dynamic workload management functions through the addition of a resource manager component.

The resource manager enables the dynamic generation of the job parallel engine configuration file. Therefore, this functionality allows the dynamic assignment of jobs to available machine nodes that correspond to the selected workload management algorithm. The workload algorithm can also be configured based on the features of the resource manager software being integrated with, for example, load on that node at the time of run, job priority, time of day, or combinations of these or other customizable decisions.

The following examples are of Resource Managers that were deployed into InfoSphere Information Server grid environments:

- ▶ IBM Platform Load Sharing Facility (Platform LSF®)
- ▶ IBM Tivoli Workload Scheduler Load Leveler (Load leveler)
- ▶ Oracle Grid Engine (OGE, formerly Sun Grid Engine SGE)

The choice of workload manager can be driven by customer preference.

Within a grid environment, the workload management links to the InfoSphere Information Server to dynamically generate the nodes to use in the parallel job configuration file.

The workload manager is able to track which nodes are up or down, so a node that is down does not cause a service outage.

The resource manager component has agents that run on each server, and a master coordinator from which a request for resources are served. The master coordinator is typically located on the head node, as shown in Figure 2-20.

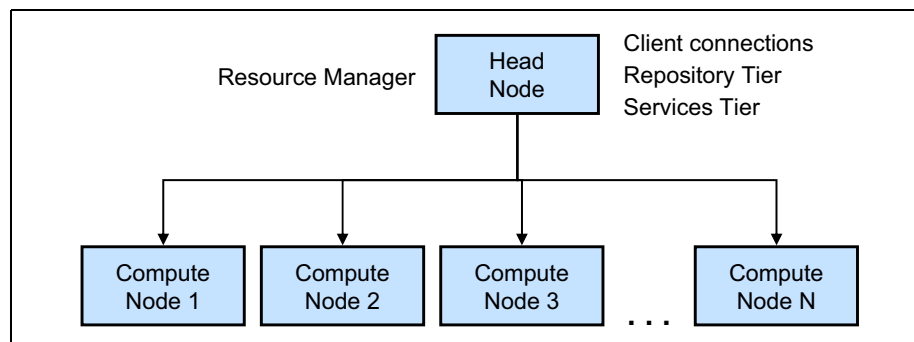


Figure 2-20 Grid head and compute nodes

Benefits and considerations of grid topology

The benefits are indicated in the following list:

- ▶ Offers massive scalability of the engine tier.
- ▶ Significant infrastructure consolidation is possible, compared to silos for departmental servers which have high idle time.
- ▶ Offers customizable workload management.
- ▶ Provides a shared service or a virtualized platform- a clear separation of the platform from customer projects.
- ▶ Offers the possibility to develop project-charging models.
- ▶ Multiple environments and projects can share a pool of compute nodes.
- ▶ Administration and complexity scales more linearly compared to MPP.
- ▶ Encourages organizational rather than departmental standards.
- ▶ Compute node pool is inherently highly available; loss of a compute node does not cause a complete service outage.
- ▶ Compute nodes can be added without a service outage.
- ▶ Multiple environments (development, test, and preproduction) can share a pool of compute nodes with workload management to make the most effective use of the available hardware resources at any point the project lifecycle.

The considerations are indicated in the following list:

- ▶ Ensure that a fast private TCP/IP network for internal node connections is available.
- ▶ As with parallel engine standard practice, use the fastest disk for parallel engine node-specific scratch areas. For example consider using a solid-state drive for the parallel-engine sort and buffer areas, as the budget allows.
- ▶ Use fast disk for data set areas that are shared across nodes.
- ▶ Cluster the head node with a compute node for head node high availability by using cluster software.
- ▶ NAS or SAN storage can be used for shared storage areas including InfoSphere Information Server installation directories. SAN storage does not inherently include cross server file sharing protocols.
- ▶ For SAN storage, which does not inherently include cross-server file-sharing protocols, cluster file-system technology is required for data set and shared data areas. IBM General Parallel File System (GPFS) or Red Hat Global File System (GFS) are examples.

2.7.9 Multiple dedicated engines topology

This topology, shown in Figure 2-21, consists of multiple engine installations where each engine is installed on its own server; however, all the installations share repository and services tiers.

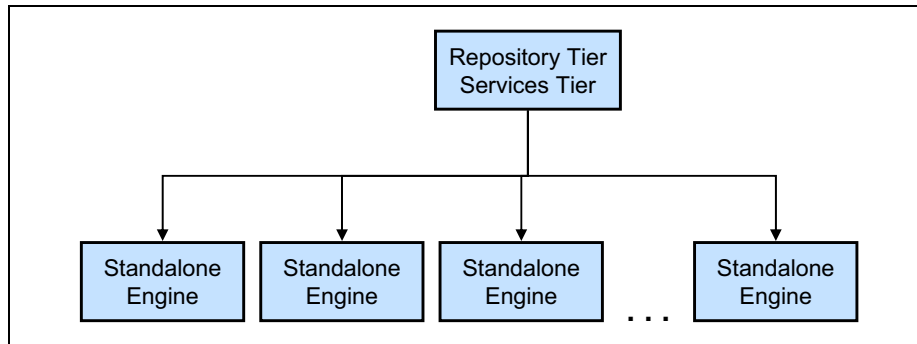


Figure 2-21 Multiple dedicated engines topology

In this case, all engine processes, including the conductor process, run on the relevant engine server. In contrast to grid topology, there is no shared storage or resource management component. Each engine installation has job configuration files, where the node names point to that engine's own server.

The primary reason for this topology is to enable a single repository and integrated governance and metadata programs, but to have complete isolation of engine resources by program to server for DataStage, QualityStage, and Information Analyzer projects. The term *program* is used here to refer to a business program which might comprise a set of projects (DataStage, QualityStage, or both) that run on their own engine installation server, completely separate from other servers. Therefore the benefit is program engine isolation, but metadata sharing.

Benefits and considerations of this topology

The benefit is complete resource isolation of parallel engine projects with the opportunity to share metadata.

The considerations are indicated in the following list:

- ▶ Workload management across engine servers is not possible.
- ▶ Engine servers each must be sized individually for peak workload, usually leading to significant aggregate over capacity in server sizing that is required for the total system.
- ▶ No inherent high availability of engine installations exists. An engine server failure will remove all projects on that server from service. Therefore, each engine installation will need to be clustered. The installation complexity therefore rises in a non-linear way with scale if high availability is required.

2.7.10 Web services engine

InfoSphere Information Server supports web services deployments, where components can respond to remote web services clients. In contrast to other architectures within a web services deployment, individual DataStage jobs are always running within enabled services, so they can respond immediately to incoming web services requests.

An example of a topology that offers both high availability and load balancing capabilities for InfoSphere Information Server web services topology is shown in Figure 2-22 on page 57. Here, the DataStage (DS) and QualityStage (QS) jobs and configuration are deployed identically to both engine tiers. Incoming requests can be routed to either leg that corresponds to the load balancing algorithm. The DataStage projects and jobs are deployed to both server legs, which have independent installations. A failure of a server in either leg means that all requests go to the remaining leg so there is no service outage, but aggregate processing capacity is correspondingly reduced for the duration of the failed server.

The front-end and services-tier load balancing is described in 2.7.2, “Clustering the services tier” on page 32.

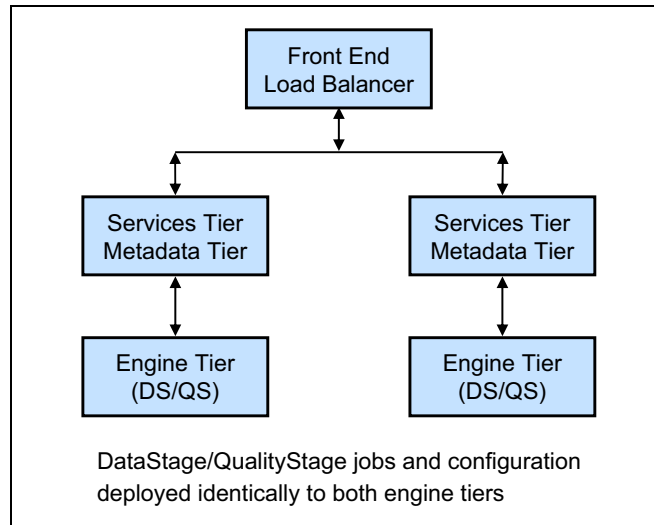


Figure 2-22 High availability web services deployment

Benefits and considerations of this topology

The benefits are indicated in the following list:

- ▶ Highly available
- ▶ Load balancing capability

A consideration is that DataStage and QualityStage projects, jobs, and rules must be deployed in synchronization across both engine installations.

2.8 Disaster recovery

It is important to distinguish *high availability (HA)* from *disaster recovery (DR)*. Prior sections in this chapter describe high availability topologies, where failure of a single component is not linked to a prolonged service outage.

Organizational disasters are considered to be power outages, fire, natural disasters, and so on, that render the entire data center inoperable, that is, all the servers are not available. In this case, having failover hardware in that data center will not assist.

The solution is for organizations to have a complete data center redundancy, at separate geographic locations.

Taking an enterprise approach to this problem, an organization can have common storage subsystems, typically SAN or NAS, that feature the ability to replicate, at the storage-system level, on behalf of the organization to duplicate storage at the second data center.

An organization might want to deploy InfoSphere Information Server within a data center that has geographic failover capability. The organization should consider that when one InfoSphere Information Server tier has a failover to another geographic site, all of the other InfoSphere Information Server tiers in that center will also fail over to that geographic site. That is, they operate in one location or the other as a suite installation.

Either SAN replication or complete level 0 (zero) file-system restore from backup into the second data center are the choices to consider for InfoSphere Information Server. SAN replication can be combined with DB2 HADR with respect to the repository tier, described in “HADR cluster” on page 39.

Following a data center failover, it is important that the new servers, which now host the InfoSphere Information Server, have the same DNS names as the original servers that host the various tiers. The InfoSphere Information Server repository has references to host names that must be maintained.

Benefits and considerations of SAN replication

A benefit is that all the replicated file systems are up-to-date at the time of data center failover.

A consideration is that, depending on the precise point of data center outage, some aspects of file systems might not be in synchronization across the tiers. As a result, after the data center failover, some amount of manual intervention might be required.

Benefits and considerations of level 0 restoration

A benefit is the opportunity to fully synchronize all the tiers that are part of the backup.

A consideration is InfoSphere Information Server data loss, from the time of the last good backup to the data center failover.

2.9 Chapter summary

This chapter describes a wide variety of deployment topologies with associated benefits and considerations. InfoSphere Information Server offers much flexibility so that organizational solution architects can make choices that match well with selected program goals, and environment and organizational platform requirements and standards.

The deployment technologies can range from a single notebook installation to a highly available Enterprise Shared Service platform with customizable workload management.



Metadata management

The discipline of information governance helps organizations to better cope with the challenges of data explosion, fast-moving markets, elevated levels of uncertainty, and the increased burden of regulatory requirements. By practicing information governance, the orchestrating of people, processes, and technology ensures that information is understood, is of high quality, and is trusted.

Metadata management is a central tenet to information governance. *Information governance* is the ability of an organization to manage its information knowledge and to answer questions such as “What do we know about our information?”

Metadata management refers to the tools, processes, and environment that are provided to enable an organization to answer the question, “How do we know what we know about our data?”

This chapter includes the following topics:

- ▶ Defining metadata
- ▶ Types of metadata
- ▶ Why metadata is important
- ▶ Where to view metadata

3.1 Defining metadata

If you search for a definition of metadata, you will find several of them. Definitions differ depending on the context domain in which the metadata is introduced. For example, metadata in the context of library management emphasizes the cataloging aspect of library materials. Libraries use metadata to capture and describe library resources and various types of publications in a manner that users, librarians, and researchers can use it to locate, sort, and trace these publications. Researchers can search for publications by subject, name, or author, or they can look for related publications about the same subject, by the same author, or in the same journal. The utilization of library resources increases many times while significantly reducing the effort involved.

In the field of real estate, metadata captures information about real estate listings. Brokers use metadata to capture the information that describes the listing, its history, and location. Buyers and sellers may use this information to research comparable listings.

In the domain of information technology (IT), metadata is about data and other IT artifacts. The most common definition of metadata is “data about data.” However, because data without context is merely a meaningless sequence of symbols, the discussions about metadata quickly revert to “information about data, information about information.” If you have a large collection of information organized in a way that you can extract relationship and data flows and perform explorations and investigation so that it becomes actionable information, you can refer to metadata as “knowledge about data.”

Maintaining a catalog of data artifacts serves a similar purpose of managing the inventory of publications in a library or property listings for a real estate broker. You need to understand what the artifacts are, trace their origins, monitor their usage, and provide services to users and stakeholders.

In the domain of information processing, the objects are electronic records of data assets. Such data assets include data files, data columns in a database, business intelligence (BI) reports, or a transformation job that moves the content of one data store into another. These data assets are the type that an enterprise creates, maintains, and uses to conduct business. Information about these data assets is most valuable to separate users within the organization. Data analysts and developers want to know the physical characteristics of these assets.

Metadata includes technical attributes that describe the physical characteristics of a data element and, often, descriptive attributes that provide semantic and administrative information, such as meaning, usage, owners, and users. This information has broad usage at all levels of an organization and by users in various roles.

3.2 Types of Metadata

Metadata has many sources, and many users find utility in various aspects of information about an object. You can classify metadata by content and usage in multiple ways. IBM InfoSphere Information Server classifies metadata into business, technical, and operational metadata types that are distinguished by their content and source.

3.2.1 Business metadata

Business metadata includes business terms and their definitions, examples of usage, business rules policies, and constraints. Altogether they define the semantics of a business concept and its realization in physical data assets.

Business metadata satisfies the needs of business people and the user community by answering the following types of questions:

- ▶ A report shows profit margin, but what does it mean?
- ▶ Where does the data for calculating the profit margin come from?
- ▶ What calculations went into the determination of the profit margin?
- ▶ Who (which *data steward*) owns this term?
- ▶ What are the business rules that are applied to profit margin?
- ▶ What other reports show profit margin?

Users of business metadata are primarily business users, but anyone can use it to understand what things mean. Examples include how, when, and by whom they are used, and what policies, rules, and restrictions might apply to the use of the data asset.

The source of business metadata is mostly the business. Subject matter experts (SMEs) and data stewards can be sources of business metadata. Internal and external business, operational, and policy manuals can be sources of business metadata.

You must express business metadata in a language that is spoken and understood by business people, which means spelling out terms fully and making sure that definitions are clear and unambiguous. Express business rules in plain language and not in a complex, formal manner with abbreviations, mathematical expressions, or functional expressions.

3.2.2 Technical metadata

Technical metadata consists of the technical description of data assets. Technical metadata includes the following descriptions:

- ▶ Schemas, tables, and file layouts
- ▶ Source and target data store identification and physical attributes
- ▶ Data quality rules
- ▶ ETL processes
- ▶ Formal specifications of transformation jobs, business rules, other processes

People and systems use technical metadata. IT technical staff, such as analysts, developers, and administrators, use technical metadata daily to perform jobs. For example, they might analyze requirements and write specifications for a new job, develop a method and write the code, or diagnose a problem and develop a fix for the problem.

Knowledge of the data and the processes that manipulate the data is valuable to businesses. Technical metadata helps to answer the following questions:

- ▶ What databases exist?
- ▶ What are the schemas, tables, views, and columns in a database?
- ▶ What are the keys of a given table?
- ▶ What do jobs read from a table?
- ▶ Can a column obtain a null value?
- ▶ What do jobs write to a table?
- ▶ What are the flat sequential files in the data stream?
- ▶ How is the data processed?
- ▶ What valid values are allowed for a column?

The source for technical metadata is usually the data assets themselves. Databases maintain data catalogs that contain all the information that a database management system needs to operate on the data that it stores. BI tools maintain data models, functions, specifications, and instructions to manipulate data and present it in the desired format.

Data integration tools maintain the details of the source, target, and flows of data, the transformations and parameters that are applied on the data, and all other details that are necessary to accomplish their designated function. All of this information that is used by the various tools to perform their tasks is technical metadata. Technical people create and use the technical metadata, and they usually store it in the individual tools. Technical people often enhance the technical data manually by annotating the data assets or by providing design and modeling documents.

3.2.3 Operational metadata

Operational metadata consists of information about the execution of an application or a job. Such information includes times and dates, counts of records processed and rejected, and statistics about processes, processors, and servers that are generated in the course of executing a job.

Operational metadata is used to answer the following questions:

- ▶ At what date and time was a job last executed?
- ▶ How many records were processed?
- ▶ How many records were rejected?
- ▶ How long did it take to process the job?

Users of operational metadata are primarily operations people who monitor the execution performance of various processes. IT and business management are often interested in the overall system performance and throughput as they consider adding more applications to existing resources or making decisions about additional computing resources. Business people might also be interested in the currency of the data that they use in their various business tasks. Therefore, they look for the last time that a job ran or that a data store was updated. Business people can obtain answers by reviewing the dates and times that a specific job was executed and that a data store was updated.

3.2.4 Metadata usage classifications

For the purposes of this book, we further identify two usage classifications of metadata. These metadata subtypes are differentiated by when and where the metadata is viewed and used by the organization. Determining how metadata is to be viewed and for what purpose it is to be used can aid organizations in determining their environment.

Design metadata

Design metadata consists of both business and technical metadata and is used during the development phase of an integration project. Design metadata is used collaboratively within the InfoSphere Information Server and reused across multiple integration tasks. Design metadata is created by InfoSphere Information Server components and includes items such as these items:

- ▶ Table analysis results
- ▶ Database table definitions
- ▶ Business terms
- ▶ Mapping specifications

Design metadata can be used by technical users who move data from source to target, or business users within the scope of an integration project.

Reporting metadata

Reporting metadata consists of all three metadata types: business, technical, and operational. The metadata is used to gain an understanding of the movement and relationships of data between the various data stores within the organization. Reporting metadata can include metadata that is created by InfoSphere Information Server components and also metadata that is imported from external sources and includes items such as these items:

- ▶ ETL jobs and execution results
- ▶ Data sources
- ▶ Business terms
- ▶ External processes and applications
- ▶ BI reports

Reporting metadata can be used within a data governance initiative to help in building trust in the accuracy and validity of data.

3.3 Why metadata is important

Metadata management is not only the cataloging of information about data objects. It also includes the ability to identify relevant metadata providing ability to enrich it, maintain its currency, and persist the metadata in such a way as to encourage its reuse. Metadata management provides the tools, processes, and environment to enable an organization to answer the question, “How do we know what we know about our data?” The capability to easily and conveniently locate and retrieve information about data objects, their meaning, physical location, characteristics, and usage is powerful and beneficial to the enterprise. This capability enhances the ability of the organization to deal with risk, meet regulatory requirements, and improve IT productivity.

3.3.1 Risk avoidance through trusted information

Organizations face a multitude of risk types of which market, operational, and regulatory exposure are only a few. Information and knowledge are ways that an organization can mitigate risk. The capability to make plans and decisions based on reliable and trusted information does not eliminate risk. However, with this ability, businesses can more precisely evaluate the risk that they face and act accordingly. As the amount of data and systems grow within an organization, data integration becomes more complex and therefore makes trusted information more important. All drivers of information governance rely on the premise that the

information they report and use to make decisions is reliable. Metadata management provides the measure of trust that businesses need. Through data lineage and impact analysis, businesses can know the accuracy, completeness, and currency of the data that is used in their planning or decision-making models.

3.3.2 Regulatory compliance

More than ever before, organizations are subject to regulatory requirements. They are required to submit periodical reports concerning their activities in their particular business domain. Certain regulatory directives are particular about data handling, specifying the security, privacy, and retention requirements on different types of data. Organizations are often subjected to audits, where they are required to show and demonstrate that they comply with these requirements. Metadata management conducted on a unified platform that provides stewardship, data lineage, and impact analysis services is the best assurance that an organization can validate and demonstrate that the data reported is true. It also validates and demonstrates that the data is correct and is handled according to regulations.

3.3.3 IT productivity

Many organizations are bogged down with the repetitive IT tasks of analyzing and reanalyzing the same data and processes when they are called to support a new service or reporting requirement. Data is abundant. Much of it comes from existing systems and data stores for which no documentation exists or, at best, the documentation does not reflect the many changes and updates of those systems and data stores. In many cases, this precious knowledge remained with the individuals who originally developed the system, but who might have left the company or moved on to roles in other departments or operations.

This lack of documentation translates to labor-intensive tasks of reverse engineering, research, and analysis to identify the elements that are required for the new tasks, their properties, and dependencies. You can avoid part of this effort if the metadata is captured and maintained in a repository that is accessible to developers, analysts, and other stakeholders.

3.3.4 Requirements for managing metadata

The key to metadata management requires a committed partnership between business and IT with joint ownership or guardianship in which both communities contribute time and effort and have input to the final product look and function.

Metadata management initiatives usually fall within the realm of the broader information governance program. Metadata management is an enabling practice, supporting management in its daily chores by providing trusted information for reporting, planning, and decision making. Metadata management helps users assert the meaning of information and determine its accuracy, completeness, and currency.

For a long time, information professionals and corporate management recognized that data is a valuable asset and must be treated carefully. Although most organizations do not document the value of information assets, they are aware that they can extract value from the available information and might suffer damage if they fail to handle and secure it properly.

The discipline of information governance emerges as a primary condition for maintaining a healthy corporation. Presently, laws or regulations require many businesses and government entities to maintain specified levels of integrity, security, and privacy of their data and to be able to show compliance with these requirements. Beyond that, the ability of an organization to guarantee high-quality information or maintain the knowledge about its information is invaluable to its ability to compete successfully in today's markets.

Over the decades, information systems have evolved, creating a complex web of applications and data stores that feed each other. Transactional data systems perform monetary or material transactions, money transfers between accounts, reservations are made for trips, purchases are recorded, and shipping instructions are issued. These transactions represent a repository of valuable information that companies harvest and mine to extract patterns and identify trends.

Companies use BI applications to digest this data and present it in ways that managers and other knowledge workers can use to plan and make decisions. Businesses create information supply chains that consist of pipes, channels, and applications. IT uses various tools and utilities along these data routes to cleanse, integrate, consolidate, and reconcile data. These tools populate master data repositories, data warehouses, and data marts that are designed to meet various information needs.

At any given time, organizations deploy various information processing initiatives. Whether upgrading an application to support new offerings, products, and services, responding to compliance reporting requirements, or adopting new analytical and decision making disciplines, IT departments are busy moving information from one container to another, changing its format and representation.

In the present information environment, these tasks have many risks. A failure to deliver, and possible exposure of the organization to legal and regulatory liability, can result from the following issues:

- ▶ The failure to obtain complete and true specifications
- ▶ A lack of understanding the meaning and use of the data elements
- ▶ A lack of understanding the restriction and constraints that might apply to data elements

In traditional information management, knowledge resided in separate systems or with only individuals who had the knowledge of the subject or, if documented, on their personal workstation or in a drawer. Shared drives and directives to store all of these documents in a public area provide partial relief, but they can quickly become outdated. When companies deposit data in architectural designs, data dictionaries, system specifications, and other documents, these repositories do not lend themselves easily to search and exploration. Although information might be accessible to a broad community of users in a single location, the effort to identify the correct documents, determine their currency and relevance, and retrieve the information might be significant.

The cost and risk that are associated with this style of operation are prohibitive, and organizations recognized this issue a long time ago. System development methodologies that have been introduced over the years have made significant strides in streamlining and rationalizing the software development process. However, not much was done in support of the preservation of the knowledge generated in the process post deployment.

Metadata management is the practice of managing knowledge about the information supply chain. Although many people refer to metadata as “data about data,” in reality, companies work with more than only data or information. Metadata refers to a rich structure of knowledge. This structure captures the meaning of a term or data asset, its relationships to other assets, and rules that might apply to it to determine the quality, policies, and regulations that specify its use.

Metadata management addresses many of the challenges that organizations face in the present reality of a fast-moving world. Transactions execute in a fraction of a second, and decision-making needs to match this speed. Trusted information in this world is invaluable, and the capability to trace and track the flow of data and access the information associated with it is critical.

3.4 InfoSphere Information Server metadata

InfoSphere Information Server manages a variety of metadata assets, generated both internally and externally, allowing for an enterprise view of information. For a complete in-depth description of metadata management, see *Metadata Management with IBM InfoSphere Information Server*, SG24-7939.

3.4.1 InfoSphere Information Server internal metadata

All InfoSphere Information Server product modules automatically generate and consume metadata. The metadata that they generate is persisted in the InfoSphere Information Server repository. This behavior is true for all InfoSphere Information Server instances, such as development, test or QA, production, and other environments that a company might have.

The internally generated metadata is persisted in the shared InfoSphere Information Server repository. Assets in the shared repository are available for reuse by all components within the InfoSphere Information Server. For example, a table definition that is imported during an Information Analyzer profiling activity is immediately available for reuse by DataStage or Metadata Workbench.

3.4.2 InfoSphere Information Server external metadata

InfoSphere Information Server has the capability to import, enrich, and maintain metadata that is created by other tools. This external metadata must first be identified, harvested, and then loaded into the shared repository. This loading can be accomplished by either the Import Export Manager or the InfoSphere Metadata Asset Manager applications, described in Chapter 1, “InfoSphere Information Server overview” on page 1.

After the external metadata has been loaded into the shared repository, it becomes available to all components within InfoSphere Information Server. For example, a table definition that is created by a modeling tool can be used by Information Analyzer. A Business Intelligence report is available to Metadata Workbench to be viewed within a data lineage report.

3.4.3 Metadata lifecycle

A normal approach to migrating between application environments within the software development lifecycle (SDLC) follows well-defined steps for moving assets. However, metadata within the InfoSphere Information Server environment may follow a different approach. Metadata that is defined within a

development environment might not be required or appropriate for moving into other environments such as production. For example, operational metadata that is used to create data lineage in a development is irrelevant in production. The methodology and metadata lifecycle is described further in Chapter 4, “Lifecycle management of assets” on page 75.

3.5 Where to view InfoSphere Information Server metadata

This section describes the potential of having a dedicated metadata environment. The purpose of this dedicated environment is to provide all stakeholders full-time access to the various types of metadata that exist within an organization and is stored in InfoSphere Information Server.

3.5.1 Business requirements for deployment

Depending on business requirements, InfoSphere Information Server product modules might be deployed in some or all of the environments. For example, DataStage is always deployed in all environments because it is part of the development cycle to promote it through various environments. Metadata Workbench might not need to be installed (and used) in a development environment, but it might need to be installed in a production environment.

In the dedicated environment, regardless of the environment in which the metadata is created and persisted, the relevant metadata is copied to this separate instance of InfoSphere Information Server that is dedicated to accumulating metadata. This dedicated environment becomes the place where all stakeholders can go and view the metadata they want.

For example, the business analyst wants the resultant data to be properly formed and accurate. The data steward, or potentially the CEO, wants all of the data assets to be verifiable and conform to regulatory constraints, if they exist. The ETL developer wants to know how to efficiently and accurately develop applications that get the job done. All of these examples require distinct access to the InfoSphere Information Server.

Some organizations might oppose any access to the production servers, often with good reason. Others, also for good reason, might demand access to the production data so that they can be confident in the data that is being produced. It is being implemented by various enterprises.

When you know the most crucial decision points to consider, you can set the policy to align with corporate concerns. Start by focusing on the tasks that InfoSphere Information Server performs, such as the following examples:

- ▶ Moves data
- ▶ Cleanses or transforms data
- ▶ Analyzes data
- ▶ Stores business definitions
- ▶ Assigns owners to data assets

Then look at the users who are not just developers and operation staff in this environment. Many other groups of users in one way or another need to access the InfoSphere Information Server. Considering that they all have valid reasons, do you give them access to the production system? The answer depends on how you evaluate the needs of each group and weigh those needs against system performance.

A fundamental assumption is that you cannot let anyone affect the performance of the production system. What if you process data only at night, and the loads are incremental and run in a relatively short time. Then is it acceptable to allow access to operational metadata on that system during the day by using Metadata Workbench? Again, the answer depends. Look at other options that, although require additional hardware and software, might be the answer you seek.

3.5.2 Metadata environments

Organizations have differing requirements and needs when regarding metadata usage and sharing. When determining the type of environment to implement, or start with, an organization must consider the business objectives it is trying to accomplish. The major business objectives to consider when choosing an environment to implement are IT productivity and information governance. An example might be increasing productivity through increased collaboration, but information governance has not yet been prioritized. In such an instance, an environment that increases collaboration should be chosen. As information governance does become prioritized within the organization, an environment that increases collaboration and metadata reporting can be implemented.

This book identifies four types of metadata environments for consideration when using InfoSphere Information Server: simple, unified, reporting, and governance. These metadata environments are explored in more detail in Chapter 5, “Metadata deployment architectures” on page 169.

Simple

A simple metadata environment is defined by the installed components and the amount of user collaboration within the InfoSphere Information Server platform. In a simple metadata environment, collaboration between users who do separate tasks might be minimal. Additionally, enterprise metadata reporting requirements might not be significant. For instance, if only InfoSphere Information Analyzer is installed, collaboration between users is not required. Alternatively, if only DataStage is installed, enterprise-wide metadata reporting is not required.

Considerations for a simple metadata environment involve these questions:

- ▶ Which InfoSphere Information Server tools are installed?
- ▶ Will users be sharing metadata to perform their tasks?
- ▶ Will there be expectations of a heavy load related to metadata reporting?

Unified

A unified metadata environment is also defined by the installed components and the amount of user collaboration within the InfoSphere Information Server platform. In a unified metadata environment, there is a high amount of collaboration between users performing similar or related tasks. Examples include the business definitions and data analysis that are captured in InfoSphere Business Glossary, or the mapping specifications that they create in InfoSphere FastTrack. Operational metadata is also of interest to many.

Considerations for a unified metadata environment involve these questions:

- ▶ How will this infrastructure be created so it looks and feels like production, but does not affect the production environment?
- ▶ What if we replicate the parts of production that satisfy the metadata or governance needs?

Reporting

A reporting metadata environment combines metadata from a collaborative development environment with the operational metadata from a production environment, but is segregated in terms of hardware so as to not affect the production environment. Reports based on InfoSphere Information Server metadata might require a distinctive metadata reporting environment.

Considerations for a reporting metadata environment involve these questions:

- ▶ Are the enterprise metadata reporting needs affecting the development of production environments?
- ▶ Does the enterprise have a large, intricate glossary?

Governance

A governance metadata environment uses the reporting and governance functionality of InfoSphere Information Server when DataStage and QualityStage are not deployed. A governance environment is similar to the reporting environment in the amount of metadata reporting done.

Considerations for a governance metadata environment involve these questions:

- ▶ Which InfoSphere Information Server tools are installed?
- ▶ Does the enterprise require metadata reporting?



Lifecycle management of assets

This chapter describes methodologies to implement lifecycle management of InfoSphere Information Server assets. This includes source code control management for IBM InfoSphere DataStage and QualityStage applications that use IBM InfoSphere Information Server Manager and external source code control systems. In addition, this chapter describes the lifecycle management of IBM InfoSphere Information Analyzer, IBM InfoSphere Business Glossary, IBM InfoSphere Metadata Workbench and related metadata assets.

4.1 Introduction to assets

The asset interchange **istool** command-line interface can be used to move individual assets or large groups of assets between the metadata repositories of separate installations of InfoSphere Information Server. For example, you can move assets from a development environment to a test, production, or source control environment. You can build the asset interchange commands into scripts to facilitate the routine back-up or movement of large groups of assets. The speed of the import or export process depends on the size of the project.

Table 4-1 lists the assets that can be moved by using asset interchange.

Lifecycle management of the products that are available within the InfoSphere Information Server suite varies based on the product within the suite. For example, typically the DataStage and QualityStage products are deployed in the development, test, and production environments, but the Information Analyzer product may be deployed in development and production environments only. The methods that are applicable for each specific product are described in their own sections of this chapter.

The word *assets*, in the chapter title, refers to a collection of related DataStage and QualityStage, or Information Analyzer, Business Glossary, or Metadata Workbench assets. The details of these assets are described in their respective sections of this chapter.

Table 4-1 Assets that can be moved by using asset interchange

Category	Assets
IBM InfoSphere Business Glossary	<ul style="list-style-type: none">▶ Categories▶ Terms
IBM InfoSphere FastTrack	<ul style="list-style-type: none">▶ Mapping components▶ Mapping compositions▶ Mapping specifications▶ Project templates▶ Projects▶ Role assignments, reports, common metadata, glossary assets, and IBM InfoSphere DataStage and QualityStage assets that are associated with a project

Category	Assets
IBM InfoSphere Information Analyzer	<ul style="list-style-type: none"> ▶ Projects, including the assets that are associated with them, such as analysis results, data rules, and data classes ▶ All data classes, regardless of which project they are associated with ▶ Reports and common metadata that are associated with a project
IBM InfoSphere DataStage and QualityStage	<ul style="list-style-type: none"> ▶ Custom folders (external files in the project directory) ▶ Data connections ▶ Data elements ▶ Data quality specifications (lookup tables, rule sets, and match specifications) ▶ IBM IMS™ databases ▶ IMS viewsets ▶ Jobs (mainframe, parallel, sequence, and server) ▶ Job executables ▶ Machine profiles ▶ Parameter sets ▶ Routines (mainframe, parallel, and server) ▶ Shared containers (parallel and server) ▶ Stage types ▶ Table definitions ▶ Transforms ▶ Shared tables and related common metadata assets that are associated with table definitions

Category	Assets
Common metadata assets	<p>Business intelligence (BI) assets:</p> <ul style="list-style-type: none"> ▶ BI models ▶ BI collections ▶ Cubes ▶ BI reports ▶ BI report queries <p>Implemented data resources:</p> <ul style="list-style-type: none"> ▶ Host computers ▶ Databases ▶ Database schemas ▶ Stored procedures ▶ Database tables ▶ Data files ▶ Data file structures ▶ Data connections ▶ Data item definitions <p>Logical data model assets:</p> <ul style="list-style-type: none"> ▶ Logical data models ▶ Logical entities ▶ Logical relationships ▶ Entity generalization hierarchies ▶ Logical domains ▶ Subject areas ▶ Diagrams <p>Physical data model assets:</p> <ul style="list-style-type: none"> ▶ Physical data models ▶ Design tables ▶ Design stored procedures ▶ Physical domains <p>Miscellaneous assets</p> <ul style="list-style-type: none"> ▶ Data connections ▶ Custom attributes ▶ Contact libraries
Reporting assets	<ul style="list-style-type: none"> ▶ Reports ▶ Report results
Security assets	<ul style="list-style-type: none"> ▶ Users, with or without roles, credentials, and credential mapping ▶ Groups, with or without roles

Lifecycle Management for DataStage and QualityStage applications

This section of the chapter describes the lifecycle management methodologies for DataStage and QualityStage assets using IBM InfoSphere Information Server Manager and IBM Rational ClearCase® source code control system as the external source control system. The InfoSphere Information Server Manager is a deployment tool that helps you manage your jobs and related assets. In a typical environment, you implement your data transformation solutions across several systems, and control the promotion of assets between these systems. At its simplest, your environment might contain development, test, and production systems. The deployment tool is installed on a Windows computer, and you can connect to all of your systems from the same manager.

The word *assets*, in the context of this section, refers to a collection of related DataStage and QualityStage assets. Furthermore, assets in this context refer to the jobs, shared containers, table definitions, and so forth that belong to DataStage projects.

4.2 Introduction to source control and deployment

The main purpose of this section is to introduce the support for source code control and deployment that is part of InfoSphere Information Server. It expands the description in the InfoSphere Information Server documentation set by recommending a pattern for application of the source code control integration. This pattern helps you to correctly use the technology for the fastest and most efficient development and deployment of their applications. The source code control and deployment is described in the context of traditional software development environments, which is typically the case when using an integrated development environment (IDE) such as Microsoft Visual Studio or Eclipse.

We then draw the contrast with the InfoSphere Information Server. Although both approaches are *software-driven*, development and deployment fundamentally differ, specifically because of the uniqueness that InfoSphere Information Server offers through its InfoSphere Information Server repository tier, where all application design assets are stored. All development and deployment work always takes place within InfoSphere Information Server

Subsequent sections provide details of how a source code control tool (such as IBM Rational Team Concert™, ClearCase, or CVS) is integrated with InfoSphere Information Server, how to deploy, and practical advice of how to structure DataStage projects to support effective source code control and deployment. This chapter uses IBM Rational ClearCase source code control system as the external source control system to illustrate the lifecycle management of

DataStage and QualityStage assets through the InfoSphere Information Server Manager tool.

Before we continue, it is important to introduce what we characterize as the “big picture” of the present discussion. See Figure 4-1 (in the figure, “DS” refers to DataStage, and IS refers to InfoSphere Information Server).

The two main aspects of this big picture are as follows:

- ▶ Package builds and deployments
- ▶ Source code control and versioning

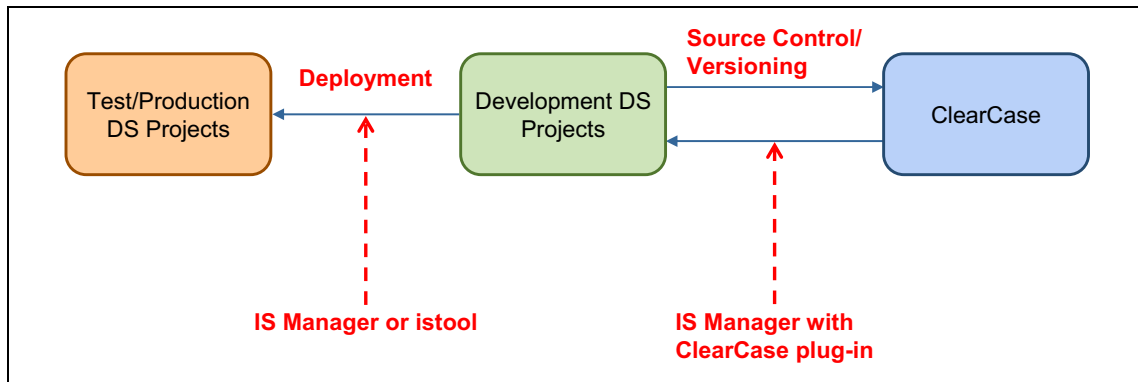


Figure 4-1 The big picture for deployment and source code control and versioning

Deployment is addressed specifically with either InfoSphere Information Server Manager or the **istool** command line utility. Source code control and versioning are addressed by the integration of a third-party tool (such as ClearCase) as a plug-in to InfoSphere Information Server Manager.

4.3 Source control and deployment in traditional software development

By traditional software development, we mean that it typically involves a stand-alone IDE, such as Microsoft Visual Studio or Eclipse. This type of environment is depicted in Figure 4-2.

Those IDEs provide the tools and conveniences, such as debugging, compiling, and building, for easier and more efficient development.

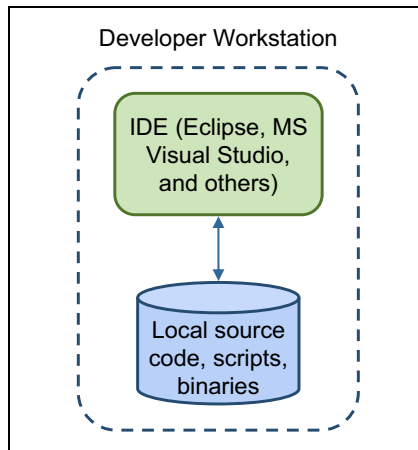


Figure 4-2 Traditional software development within a single developer workstation

In a single-developer setting, there are no concerns about coordinating updates and builds with work from other developers. Local copies of source files and assemblies are easy to maintain and manage. The only challenge is to impose self-discipline for development.

The concerns begin when having to coordinate the work among multiple developers, as depicted in Figure 4-3 on page 82, such as how to move source files and assemblies, how to maintain synchronization of work among individuals, and so forth. IDEs continue to provide for local workspaces, which rely on local copies of source files from an individual. However, IDEs do not provide mechanisms for source code control within.

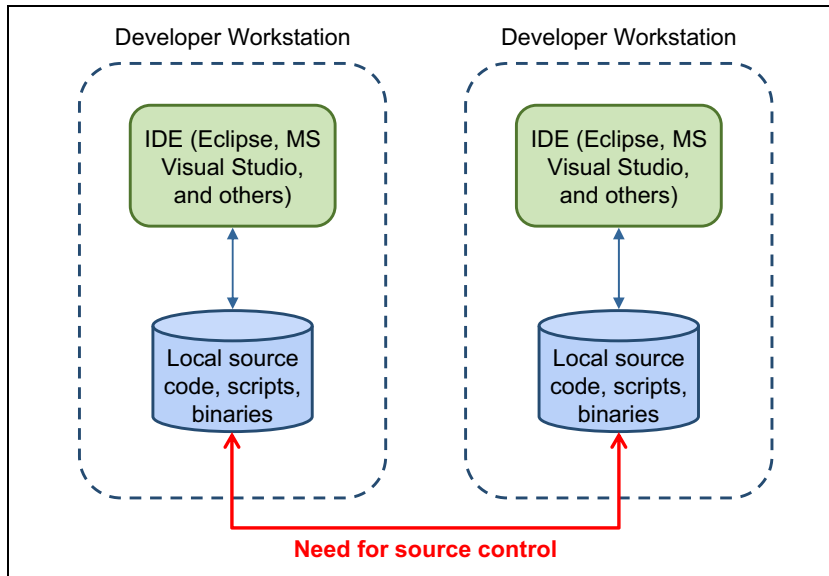


Figure 4-3 Need for source code control with multiple developers

The solution for coordinating the teamwork in an IDE-based development environment is the integration of a source code control tool, such as ClearCase or CVS, as depicted in Figure 4-4 on page 83. Virtually all enterprise class IDEs that are currently available support this type of integration.

An important aspect is that a source code control tool becomes the de facto repository for all source code and versioning. The nature of an IDE is local availability of source code files and libraries. IDEs continue to work the same way as in an individual setting, for the most part. The editing, compilation, and building processes remain the same.

Developers may create as many workspaces as they want, without incurring any administrative overhead. As long as enough local disk space is available and the source files are checked in and out of the source control repository, the developer may do whatever is preferred.

An advantage of using IDEs is that they provide menus for checking files into and out of the source code control tools.

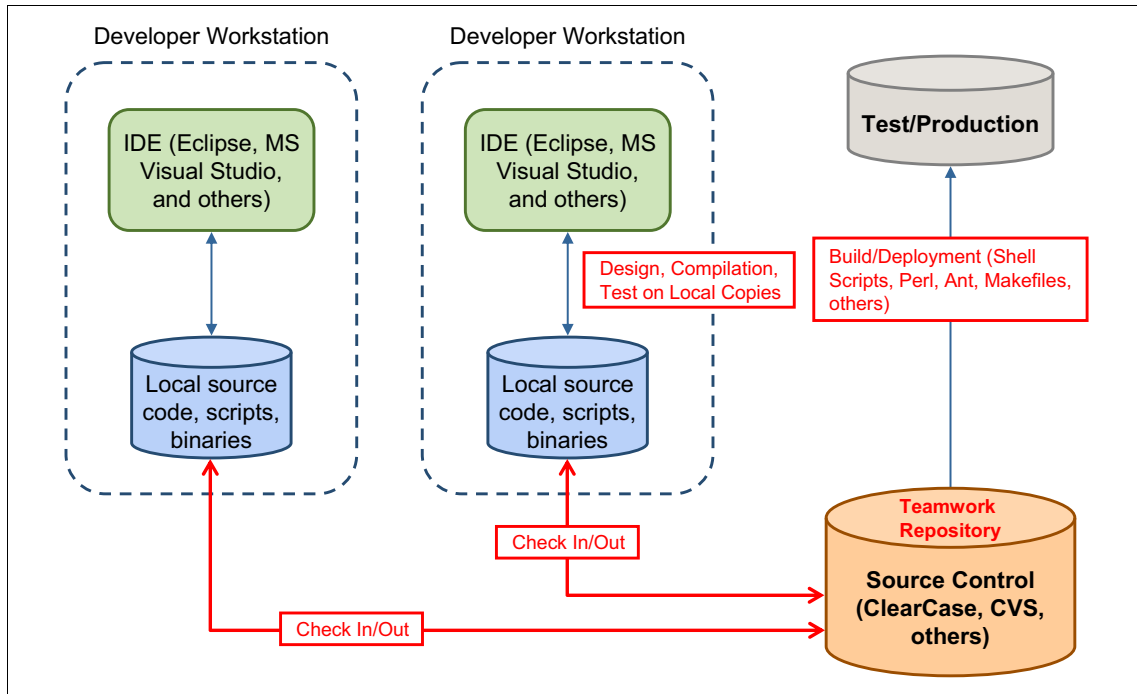


Figure 4-4 Source control as a teamwork repository

In a multi-developer setting, the topic of a *release manager* surfaces. The release manager is the person who is responsible for pulling specific versions out of the repository to create one specific maintenance release, that is, the package of assets that will be promoted from one environment to another.

Typically, versions are labeled according to a certain standard, which can make the job of the release manager much easier. After those source file versions are pulled into a *build environment*, the actual build and test process usually continues independently of the work that is performed in the individual workstations.

4.4 Source control and deployment in InfoSphere Information Server

In the InfoSphere DataStage and QualityStage environment, several notable differences are related to the managed infrastructure that these tools provide. These differences are then reflected through the development, build, and test processes, as depicted in Figure 4-5.

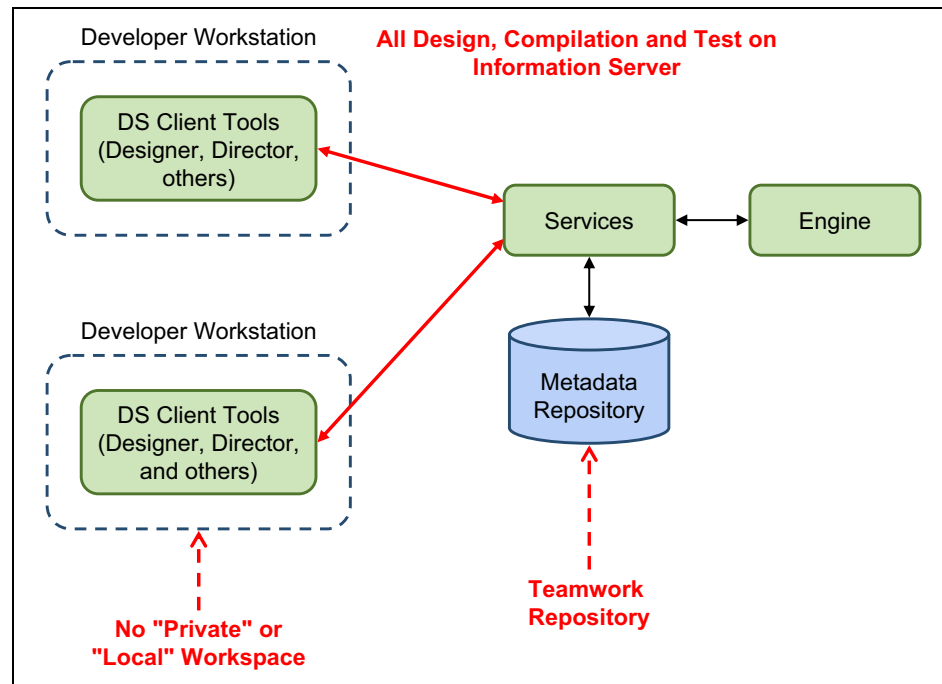


Figure 4-5 Development with InfoSphere Information Server

In DataStage, all work is done directly on the InfoSphere Information Server tiers and ultimately on the same InfoSphere Information Server repository. There is no *local workspace* that is private to each developer workstation. DataStage design artifacts are stored in that InfoSphere Information Server repository (in DataStage projects) and all compilation and testing always takes place within InfoSphere Information Server, involving some or all tiers (services, InfoSphere Information Server repository, and engine).

This approach provides the organizations with specific advantages because it promotes collaboration across both developers and business users, and provides for common backup and maintenance administration processes.

The creation of DataStage projects is a task that requires the involvement of a DataStage administrator. This task requires at least making sure enough disk space exists for the expected project size and that permissions, ownerships, and group memberships are correctly set, and so forth.

The project is a shared and collaborative environment that typically correlates to a business initiative or the development team of a department. It is unusual to create one individual DataStage project for each developer or user.

To facilitate the multi-developer nature and to provide for discrete work areas for each developer, the common approach to development in DataStage involves the following features:

- ▶ Divide the work for each developer into a separate DataStage project folder.
- ▶ Keep different versions for the same *conceptual job* design by appending suffixes.

Work is also structured in terms of folders to distinguish between major releases and *incremental fixes*. This approach is traditional and is also considered standard and suggested, as described in subsequent sections.

The integration of a source code control tool, such as Figure 4-6 on page 86 shows, simplifies the management of different asset versions, and the check in and check out process. Instead of relying entirely on suffixes, developers can simply check in and check out different versions, and keep a single version in DataStage.

However, to simultaneously work with various versions for the same job, there must continue to be different suffixes to make the distinction within a DataStage project in any given instant.

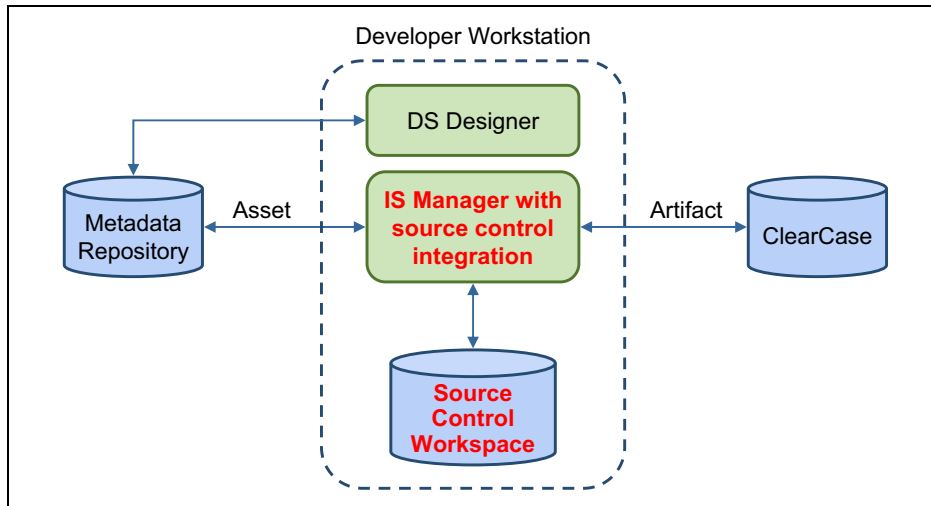


Figure 4-6 InfoSphere Information Server Manager with Source Control Integration

The integration of a source code control tool with InfoSphere Information Server takes place specifically through InfoSphere Information Server Manager.

InfoSphere Information Server Manager supports this type of integration through an Eclipse Team Plug-in (such as IBM Rational Team Concert or ClearCase Remote Client). The Eclipse Team Plug-in construct is built around the notion of a workspace for the exchange of assets between the source code control tool and another software vendor tool. The integration approach follows a standard model, using facilities that source code control vendors built.

The development work is still done through DataStage Designer against the InfoSphere Information Server repository (and the services and engine tiers also, which are abstracted in this description).

There are no menus or options available within DataStage Designer to distinguish and control the check in and check out process. All check in and check out must be done through InfoSphere Information Server Manager.

InfoSphere Information Server Manager serves the purpose of both a deployment tool and source code control.

The deployment aspect takes advantage of InfoSphere Information Server Manager features, which are relatively independent from the presence of a source code control tool. All deployment packages are always created from the InfoSphere Information Server repository. They are never created directly from the source code control repository.

To make sure the correct assets are put in a certain deployment package, the DataStage developers must make sure those are the assets present in the InfoSphere Information Server repository at the time the deployment package is built.

Figure 4-7 illustrates how various developers interact with both the InfoSphere Information Server repository and the source code control tool (in this and previous sections, the services and engine tiers are left out for simplicity).

The InfoSphere Information Server repository remains fully independent of the source control tool. The entire information about version trees, labels, tags, and so forth is contained entirely within the source code control repository.

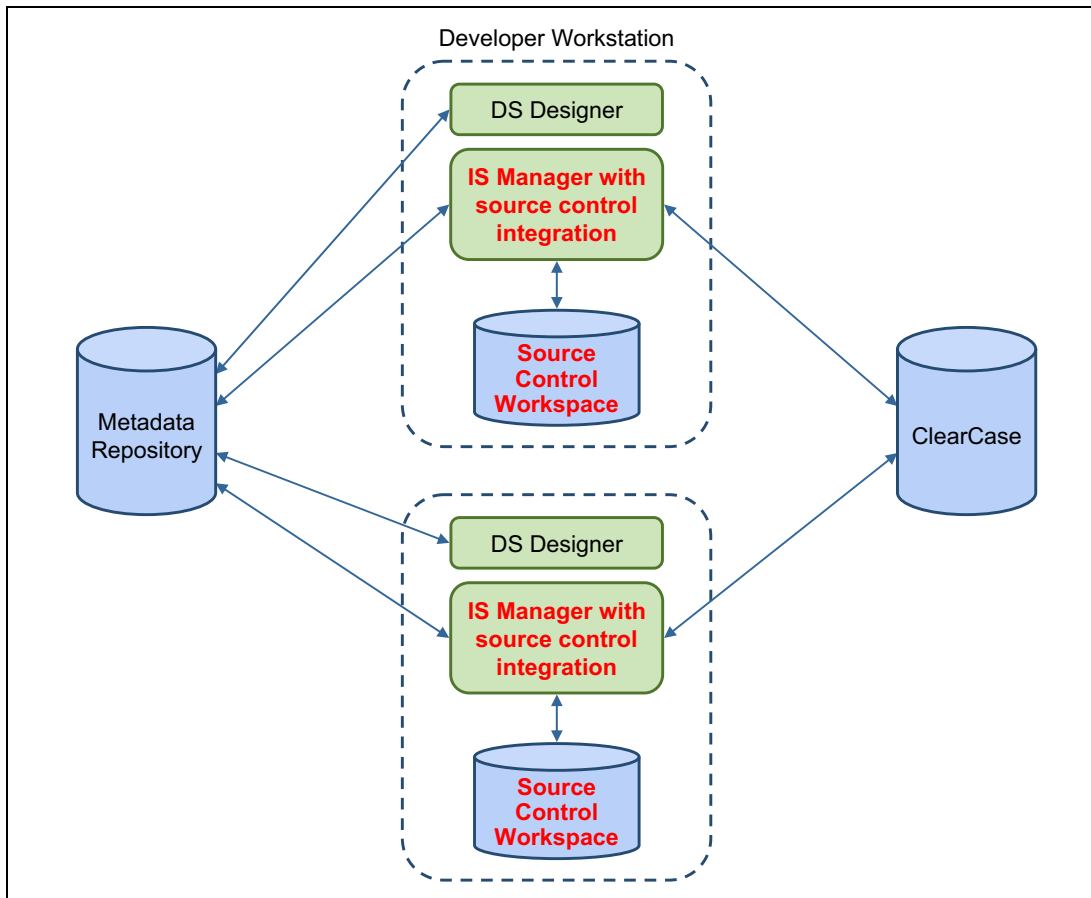


Figure 4-7 Multiple developers and source control integration

4.5 Deployment scenarios and DataStage project structures

The discussion in 4.4, “Source control and deployment in InfoSphere Information Server” on page 84 established the background of the suggestions for how to structure the work within DataStage. DataStage developers do not have development workspaces entirely private, as in the case of traditional IDEs.

All the design assets, testing and execution, and so forth always takes place directly on the server.

So an inevitable outcome is to require DataStage projects to be structured in certain ways, to address the multitude of developers working simultaneously on the same application.

The good news is that the same good practices adopted in previous versions of DataStage continue to be valid and totally compatible with the integration of a source control tool. The same discipline still applies, which in summary consists of organizing the work, releases, and incremental fixes in DataStage project folders.

4.5.1 Method

For the purpose of this document, two major deployment scenarios ultimately dictate the structure of DataStage project folders:

- ▶ Full release candidate (RC)
 - For each release candidate, there is a dedicated DataStage project.
 - The entire relevant content of a DataStage project that represents a certain release candidate is stored in a special folder (such as release).
 - The entire content of the release folder is included in the RC deployment package.
 - The project may contain other objects, such as working versions of jobs that are not supposed to be included in the release. Those are stored outside of the release folder.
 - There is a single release folder in each RC project
 - Developers may work on multiple RC versions concurrently.

- ▶ Incremental fixes
 - Only a small subset of the content of a release candidate is supposed to be assembled into a deployment package.
 - The content for an incremental fix is stored in a special folder (such as IncrementalFixN).
 - The entire content of the IncrementalFixN folder is included in the deployment package for a certain fix.
 - Any other objects that are stored in other folders are not supposed to be included in the incremental deployment package.
 - There may be multiple IncrementalFixN folders within the RC project

In both scenarios, the actual DataStage assets (jobs, sequences, and so forth) are extracted directly from the InfoSphere Information Server repository. The developers, release manager, or both are responsible to check out the precise versions, into the InfoSphere Information Server repository, that are to be compiled, tested, and certified for release.

The assumption is that anything that is assembled into a deployment package is a fully functional version that was verified to run successfully.

The creation of a new RC is a major event, implying that it has significant business impact or supports a significant supplementary code release, and that it requires assessment and approval from development, release management, or both. Examples include EDW_version1 or EDW_RC1 or SAP_FEED_RC2 or SAP_FEED_Phase2.

You should be aware of the following points during the creation of a new RC:

- ▶ Up to the moment an RC is released, it may be updated any number of times, its objects going through multiple versions.
- ▶ After release, if there are any changes, they either result in the creation of a new release candidate version (RCn) project, or in the release of an incremental fix. Incremental fixes are just refinements of a few pieces of code or minor enhancements.

Within a *Release Candidate Project*, the suggested practice is to use different folders to separate the objects under development and the objects to be released.

Relying on the source control tool to keep the information about what should be and should not be part of a full release candidate is not a good practice. To address the issue of what should or should not be part of RC, organize the objects in folders. This way, everyone can immediately see that an object might

still be in the development phase or should not even be included in a release candidate.

Give the name `release` to one folder. Everything contained in that folder is considered to be *hot* and should be part of a full *release candidate*.

Figure 4-8 presents a diagram with multiple DataStage projects, one for each release candidate. Within each project, there is one `release` folder, and one or more `IncrementalFix` folders. Assets in development may be further isolated in individual folders.

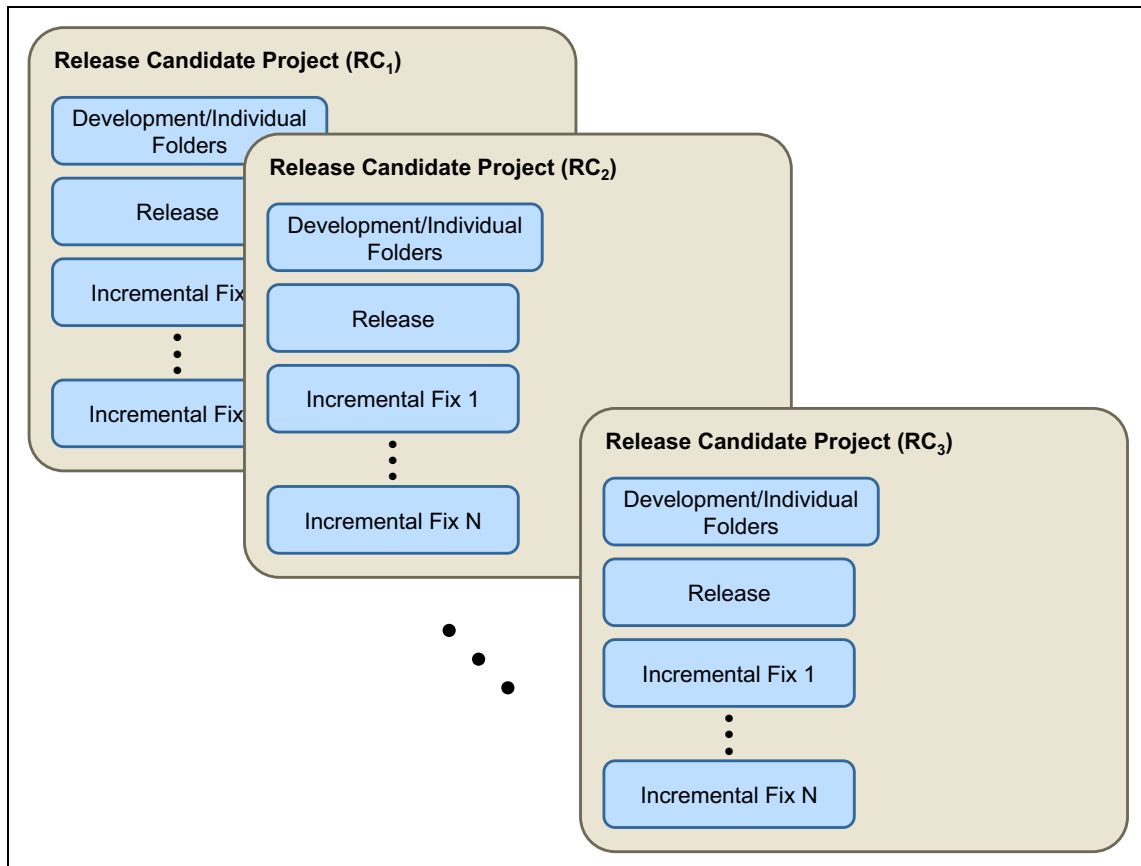


Figure 4-8 Release candidates and incremental fixes

Figure 4-9 presents the relationship between a release candidate project and the deployment and source control aspects. It also illustrates the movement of assets from one folder to another.

The presence of an asset within a certain folder clearly characterizes the state of the assets. This straightforward indication is for both developers and release managers. Instead of relying solely on source control labels or tags, this information is visible to the entire community of developers and managers.

As stated previously, these practices are the same practices that were adopted in earlier versions of DataStage and they still apply to versions 8.5, 8.7, and later.

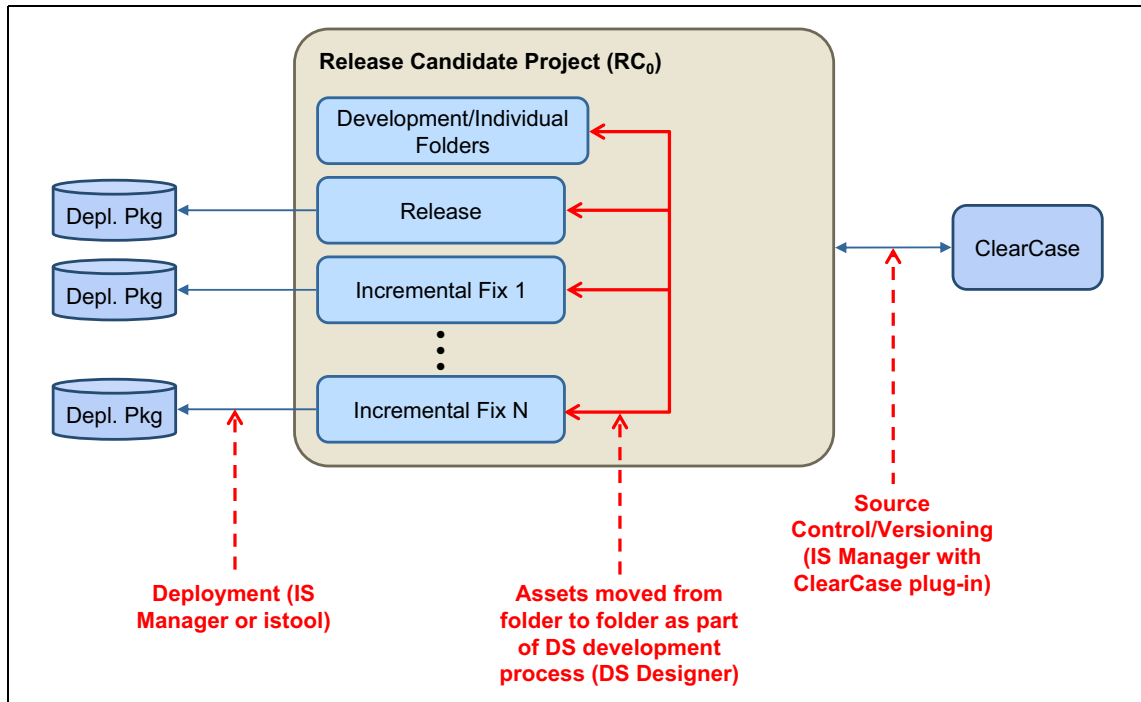


Figure 4-9 A release candidate within the context of deployment and source control

4.5.2 Method summary

The following list represents a review of several assumptions and conclusions for this section:

- ▶ Any new release candidates require the creation of a separate DataStage project. The starting point for a new release candidate, RCn, project tends to be a copy of RCn-1.
- ▶ Deployment packages are always checked into ClearCase in their entirety.
- ▶ Deployment packages are always built from DataStage either through InfoSphere Information Server Manager or istool, and not from ClearCase.
 - They always require the objects to be compiled and the entire RC to be working correctly.
 - They are extracted together, to make sure they are synchronized.
- ▶ ClearCase labels may be set as a convenience for documentation purposes, to help guide which object versions are included in the RC. However, the actual source that is put into the deployment package comes directly from the DataStage project, after you make sure it successfully executes and tests. Those objects must be removed from the release folder within the DataStage project for the release candidate itself (the same applies to Incremental fixes)
- ▶ The creation of a new RC is a major event that requires assessment and approval from development and release management.
 - Up to the moment that an RC is released, it may be updated any number of times, its objects going through multiple versions.
 - After release, if there are any changes, they either result in the creation of a new RCn project, or in the release of an Incremental fix.

4.6 Source control and versioning with InfoSphere Information Server Manager and ClearCase

A developer must use DataStage Designer to create, update, and remove job designs and other items. However, when creating deployment packages and working with versioning control, the tool for these tasks is the InfoSphere Information Server Manager.

Figure 4-10 illustrates the following steps that are involved in creating a job design and checking it into ClearCase by the developer:

1. Uses the DataStage Designer to create and update the job. After the developer decides it is in a state that worthy of submission to ClearCase, InfoSphere Information Server Manager is used for the next two steps.
2. Moves the job design to the local source control workspace.
3. Checks in the job to ClearCase.

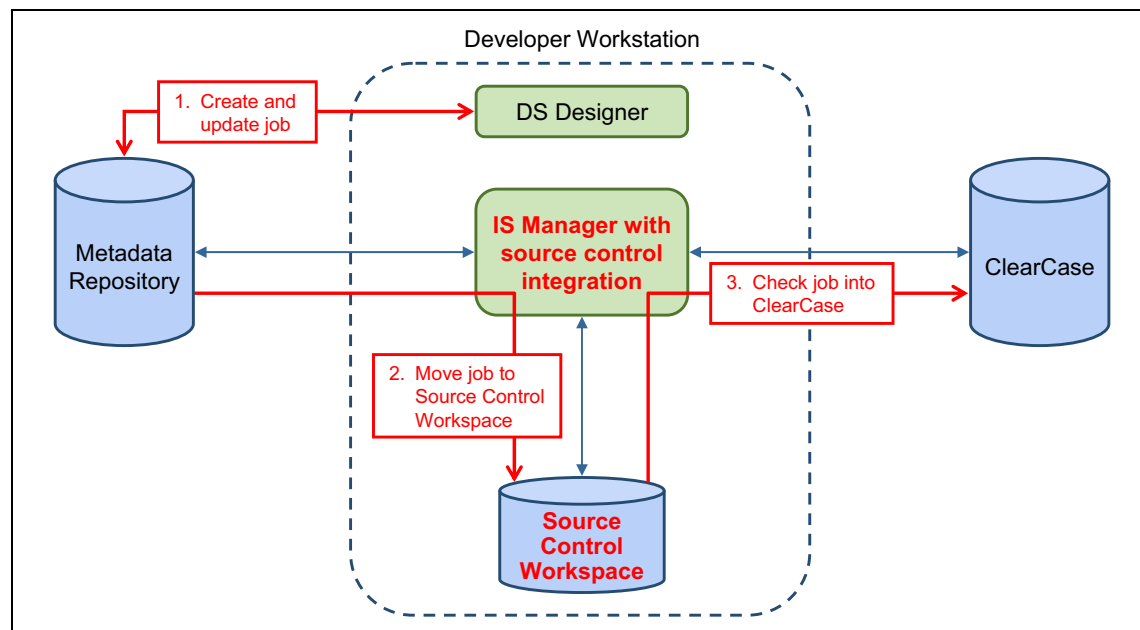


Figure 4-10 Creating and checking a job into ClearCase

Figure 4-11 expands on Figure 4-10 on page 93, showing the process of checking out a job from ClearCase, updating it through DataStage Designer, and checking the job back into ClearCase, with the following steps:

1. Using the ClearCase Remote Client Plug-in from within InfoSphere Information Server Manager, the job is checked out of ClearCase (resulting in a copy of this job being saved into the source control workspace, local to the developer workstation).
2. With InfoSphere Information Server Manager, the developer updates the InfoSphere Information Server repository with the version stored in the source control workspace.
3. With DataStage Designer, the job is updated, compiled, and tested. All updates are directly made in the InfoSphere Information Server repository. The developer may keep the job checked out for as long as needed, until the job is at a point where checking it back into ClearCase makes sense.
4. After the update step is finished, the developer moves a copy of the new version of this job from the InfoSphere Information Server repository into the source control workspace. Just as in steps 1 and 2, this task is done in InfoSphere Information Server Manager.
5. Still in InfoSphere Information Server Manager, the developer checks the job back into ClearCase through the Remote Client Plug-in.

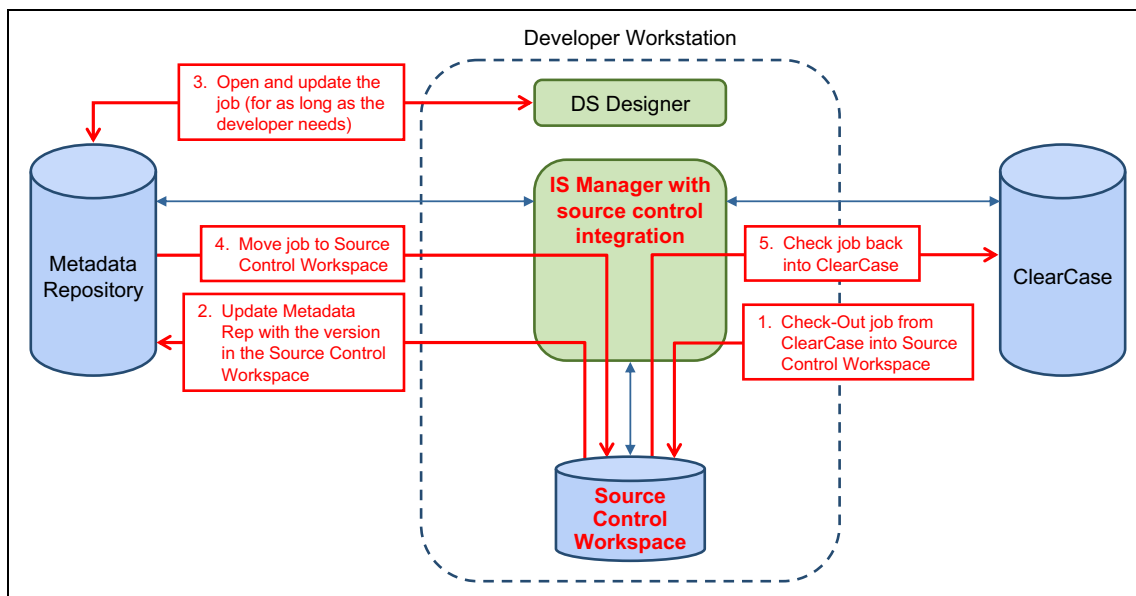


Figure 4-11 Check-In and check-out cycle

4.7 Deployment with InfoSphere Information Server Manager

Figure 4-12 illustrates the process of promoting packages through the InfoSphere Information Server Manager package editor *Send* option.

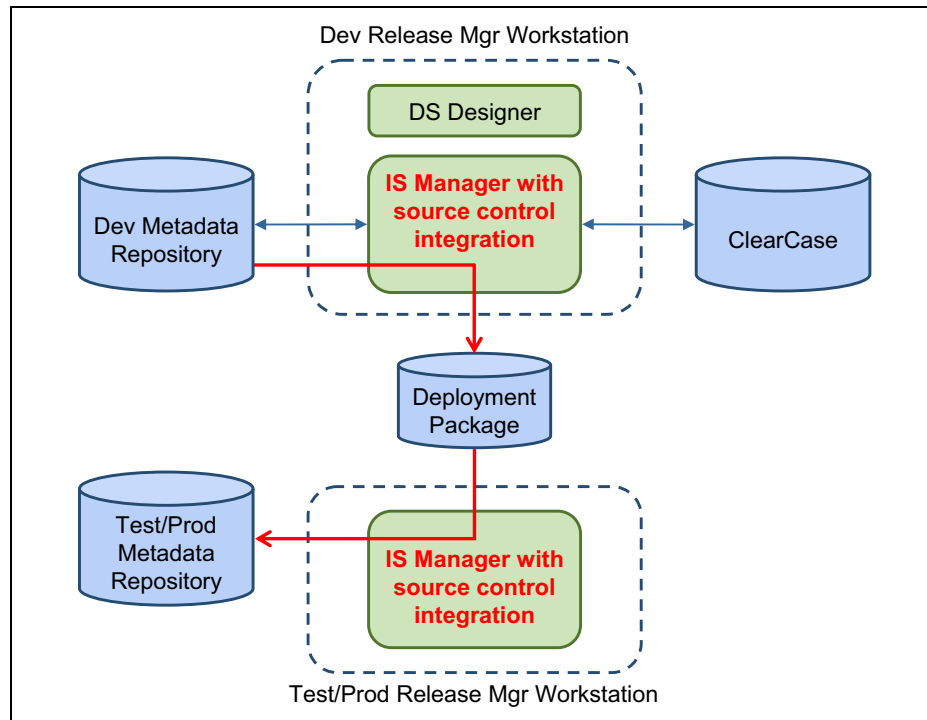


Figure 4-12 Promoting deployment packages through the *Send* option

Figure 4-13 illustrates the process of promoting packages through the InfoSphere Information Server Manager ClearCase perspective.

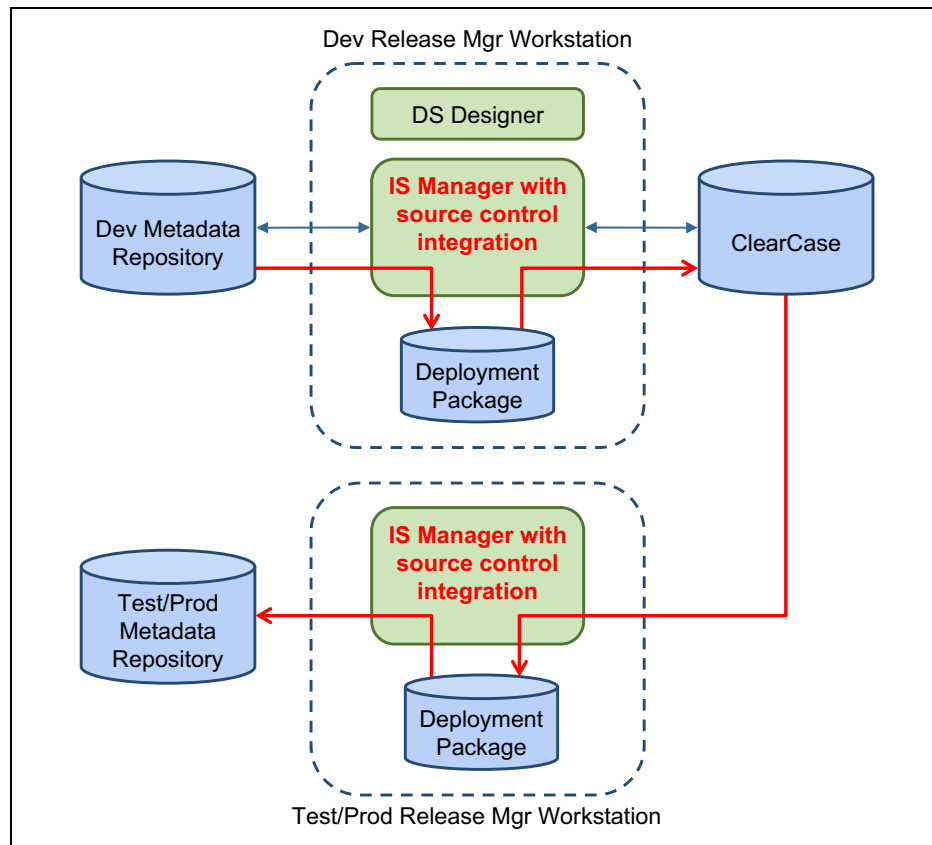


Figure 4-13 Promoting deployment packages through ClearCase check in/out

Figure 4-14 on page 97 illustrates the process of checking into ClearCase the deployment packages that are built for each release candidate version (RCn). This illustration relates to the process of creation of multiple RC versions (where each RCn version is a separate DataStage project).

The advantage of checking deployment packages into ClearCase is that release and test managers can apply comments or labels to document the progress and release for subsequent test phases or production.

This approach also ensures the ability of not only tracking the versions of the assets, but also securing a robust environment recreation mechanism that can go back to any particular backup-restore point.

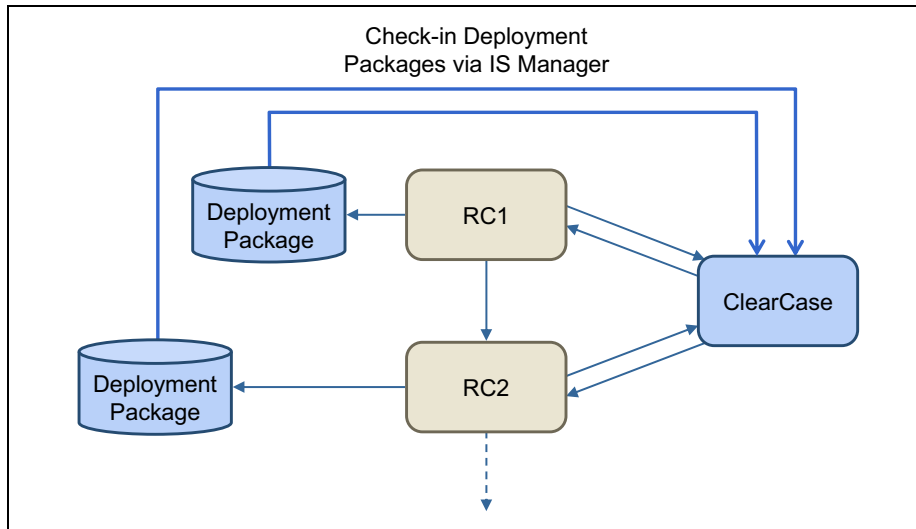


Figure 4-14 Checking in deployment packages (ClearCase-only)

This figure does not include the loading of the deployment packages into the test environments. It only serves to illustrate the use of InfoSphere Information Server Manager as the tool to check a deployment package into ClearCase.

4.8 Workflow for multiple release candidates

This section describes a scenario that involves multiple release candidates for a DataStage application, according to the assumptions listed in 4.5, "Deployment scenarios and DataStage project structures" on page 88.

A lifecycle management process can be implemented by using the development, test, and production environments for the DataStage applications.

The standard practice is to have a separate physical environment for each phase in the lifecycle. The development, test, and production systems have separate physical environments.

A mixed system, with the development and test systems sharing a single physical environment, and the production system in a separate physical environment, can also be implemented.

A suggestion is to *not* share the development, test, and production systems in a common physical environment. Also, changes should be made *only* in the development environment and then propagated through subsequent phases and environments.

A system lifecycle that incorporates more phases/environments, as indicated in the following list, is also typical at most sites:

- ▶ Development and maintenance
- ▶ Integration testing
- ▶ Quality assurance
- ▶ User acceptance testing
- ▶ Production

This chapter and subsequent chapters assume the following DataStage environments as examples:

- ▶ Development (DEV)
- ▶ Systems integration test (SIT)
- ▶ Tech test (TT)
- ▶ Production (PROD)

Figure 4-15 illustrates the workflow for multiple release candidates.

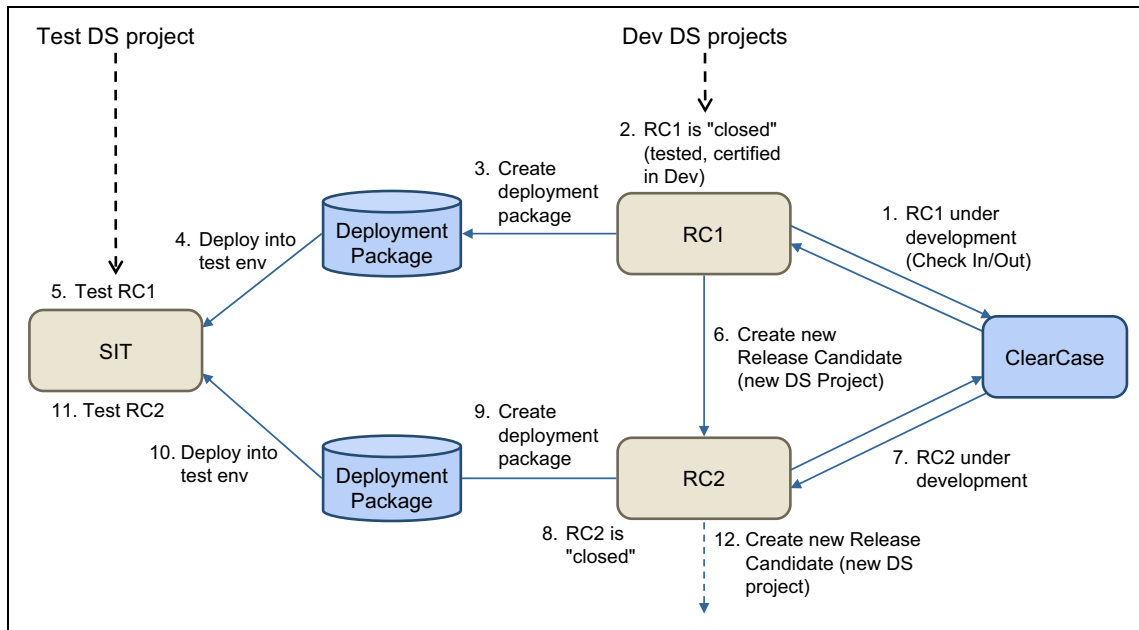


Figure 4-15 Workflow for multiple release candidates

The following steps are illustrated in Figure 4-15 on page 98:

1. RC1 is under development. Developers check objects in and out of ClearCase through InfoSphere Information Server Manager, create and drop objects as necessary, and perform individual unit tests.
2. RC1 is fully tested and certified for release to subsequent environments.
3. A deployment package for RC1 is created, by selecting from a set of standardized folders (such as the release folder).
4. The deployment package is loaded into the target test environment.
5. RC1 is tested on SIT. Steps 4 and 5 are repeated for TT, and potentially PROD if everything is successful.
6. A new RC is created (RC2). This step might be a result of required fixes and additional features and implies a new DataStage project created for RC2.
7. RC2 is under development. Developers check objects in to and out of ClearCase through InfoSphere Information Server Manager, create and drop objects as necessary, and do individual unit tests.
8. RC2 is fully tested and certified for release to subsequent environments.
9. A deployment package for RC2 is created, by selecting from a set of standardized folders.
10. A deployment package is loaded into target test environment
11. RC2 is tested on SIT. Steps 4 and 5 are repeated for TT, and potentially PROD if everything is successful.
12. A new RC is created. The cycle repeats.

4.9 Context for deployment and versioning tools

Figure 4-16 illustrates the separation between the deployment of packages onto separate testing and validation environments, and the versioning that happens within each RC development.

ClearCase is used as a versioning control solution within each project.

The creation of deployment packages is done in InfoSphere Information Server Manager (or possibly by istool utility). The packages are saved to file systems and loaded into the target test environments also through InfoSphere Information Server Manager.

In Figure 4-16, deployment packages are not checked into ClearCase, as in the scenario that is described in 4.8, “Workflow for multiple release candidates” on page 97.

The creation of separate DataStage projects for each release candidate conforms to the assumptions listed in 4.5, “Deployment scenarios and DataStage project structures” on page 88. The creation of a new release candidate is a major event that implies the creation of a new DataStage project.

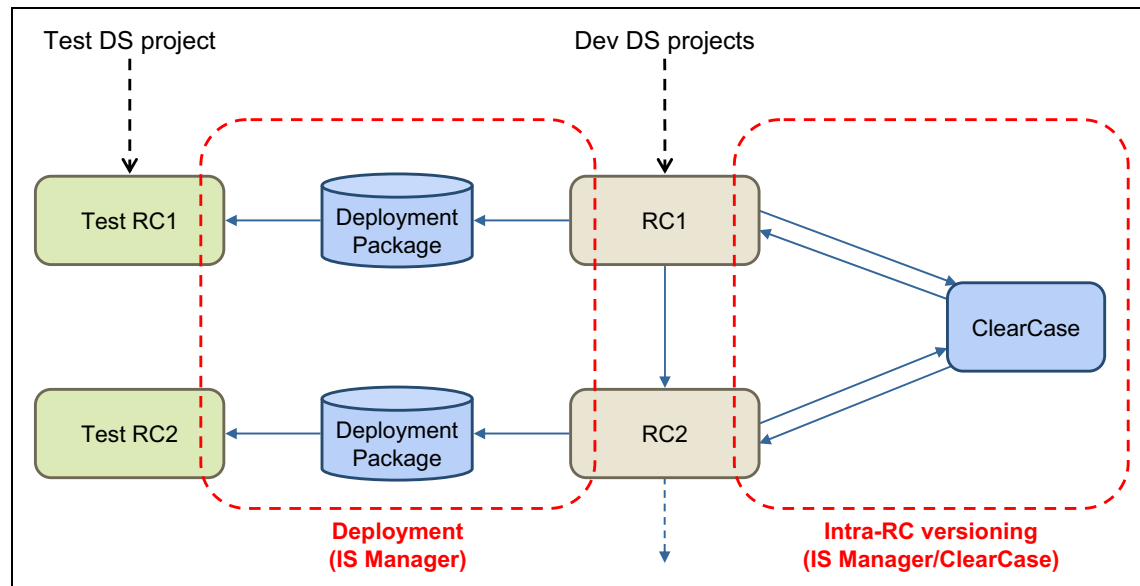


Figure 4-16 The contexts for InfoSphere Information Server deployment and versioning tools

4.10 Workflow for promotion into production

Figure 4-17 presents the high-level workflow for a deployment package that is successfully promoted and tested across the various environments, from development to production.

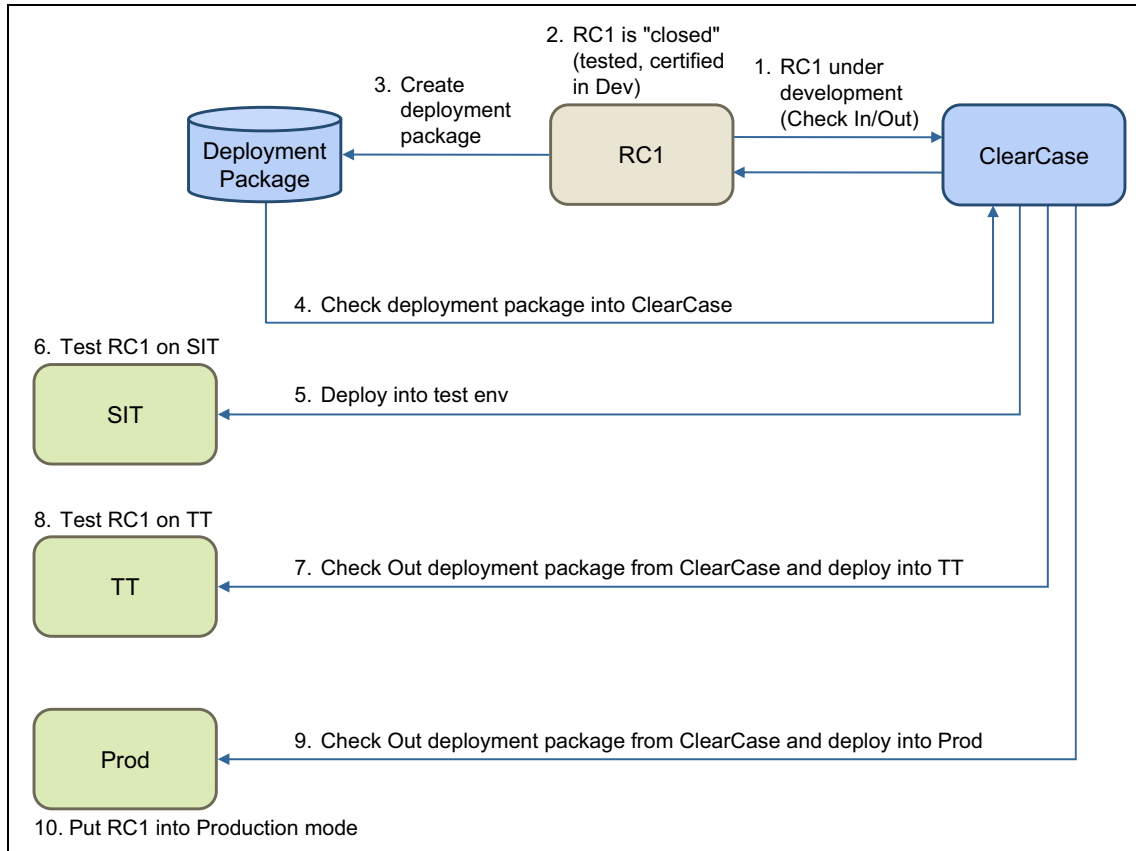


Figure 4-17 Workflow for release candidate put in production

The implementation of this process that uses InfoSphere Information Server Manager and the ClearCase Eclipse plug-in is described in 4.20, "Creating and building deployment packages" on page 149 and 4.22, "Promoting deployment packages with ClearCase Explorer" on page 155.

The following steps describe Figure 4-17 on page 101:

1. RC1 is under development. Developers check objects in and out of ClearCase through InfoSphere Information Server Manager., create and drop objects as necessary, and do individual unit tests.
2. RC1 is fully tested and certified for release to subsequent environments.
3. A deployment package for RC1 is built. This step includes DataStage objects (jobs, sequences, parameter sets, and so on).
4. The deployment package is checked into ClearCase.
5. The package is checked out and deployed into SIT.
6. RC1 is tested on SIT.
7. Testing is successful and annotated by the SIT manager in ClearCase. The RC1 deployment package is checked out of ClearCase and deployed into TT.
8. RC1 is tested on TT.
9. Testing is successful and annotated by TT manager in ClearCase. Its deployment package is again checked out of ClearCase, and loaded into PROD.
10. RC1 is put into production mode.

4.11 Build environment

The compilation of DataStage jobs is a process that requires the involvement of a DataStage client system. The compilation takes place mostly on the DataStage Designer side, although the Designer naturally interacts with the services, engine, and InfoSphere Information Server repository tiers during the compilation. Therefore, nightly batch builds must be accomplished either through existing individual developer or administrator workstations, or by allocating a dedicated machine in a shared environment (such as Citrix) for this purpose.

The compilation might be triggered by one of the following approaches:

- ▶ The command-line compiler, which is `dsc.exe`
The `dsc.exe` accepts folders as arguments, however subfolders are not included: all subfolders must be referenced explicitly.
- ▶ The batch compiler, which is accessible through DataStage Designer
The batch compiler in DataStage Designer provides the extra convenience of selecting entire folders, including all its subfolders.

Therefore, a standard practice is to adopt the batch compiler in DataStage Designer for the nightly builds, invoking this tool from a machine in a shared Citrix environment. This machine should be located as closely as possible to the InfoSphere Information Server instances, because of the heavy network traffic inherent to this type of activity.

4.12 Connecting with InfoSphere Information Server Manager to a domain

This section explains the process of establishing a connection to an InfoSphere Information Server domain. The figures in this section (Figure 4-18 on page 104 through Figure 4-21 on page 107) illustrate the following information:

- ▶ How to locate the InfoSphere Information Server Manager icon on the Windows desktop
- ▶ How to add a new domain definition, enter user credentials, and test the connection
- ▶ What the appearance of the Repository view is after a successful connection to a domain

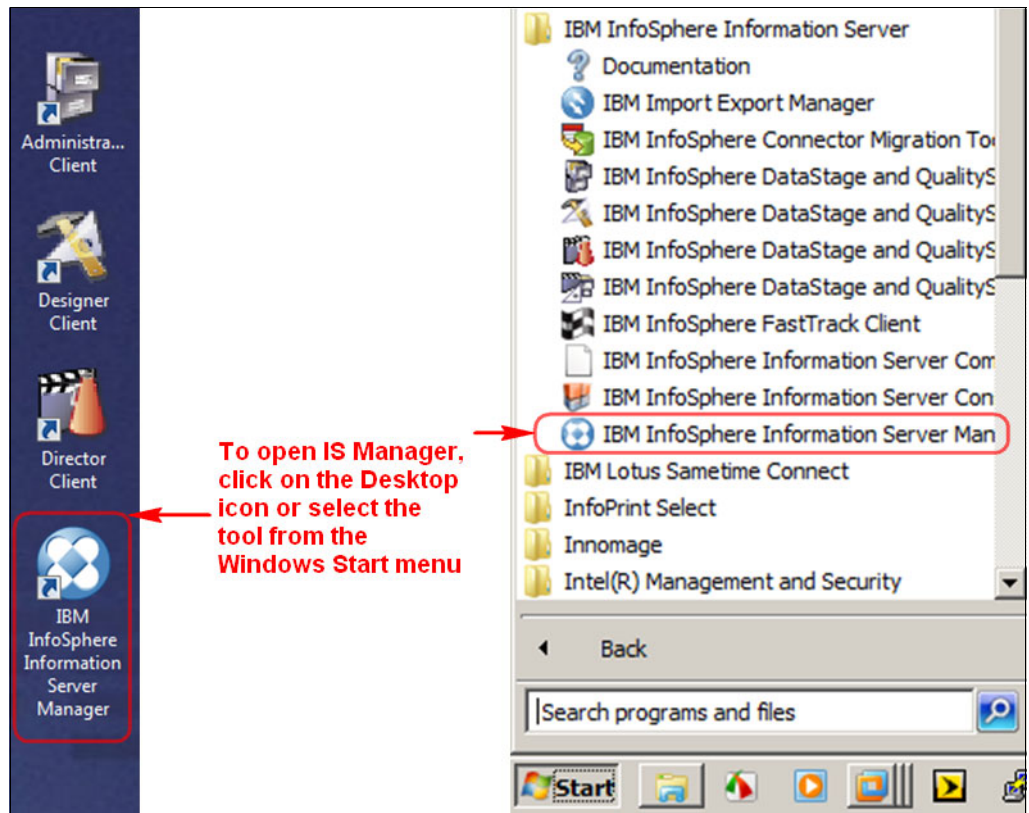


Figure 4-18 Locating the InfoSphere Information Server Manager icon on the Windows desktop

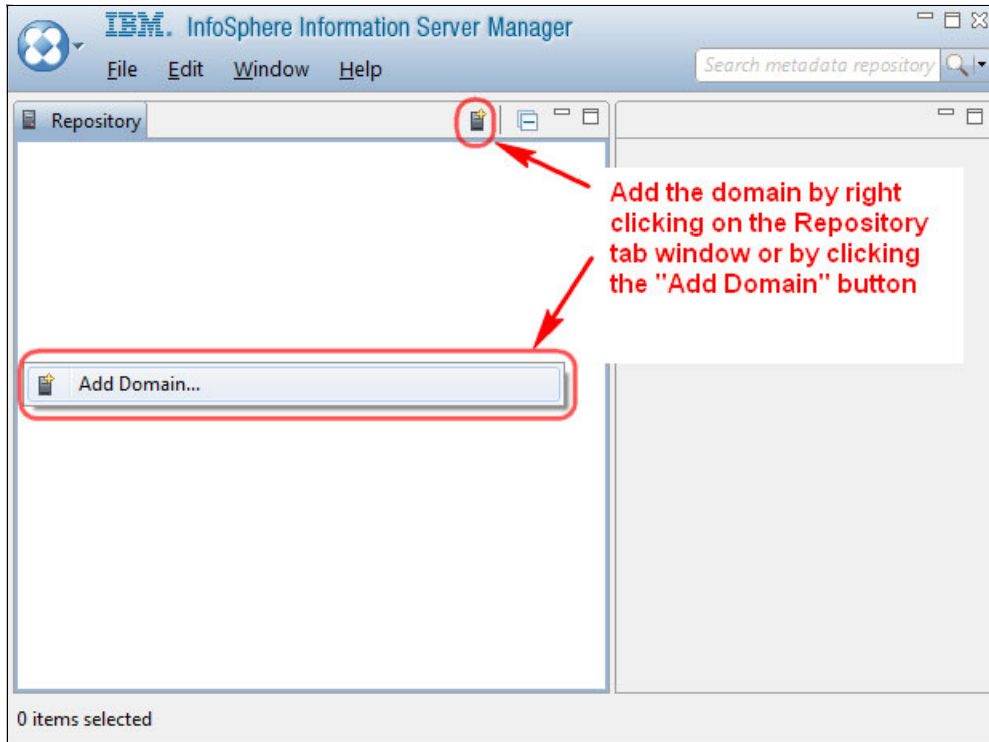


Figure 4-19 Selecting the Add Domain option

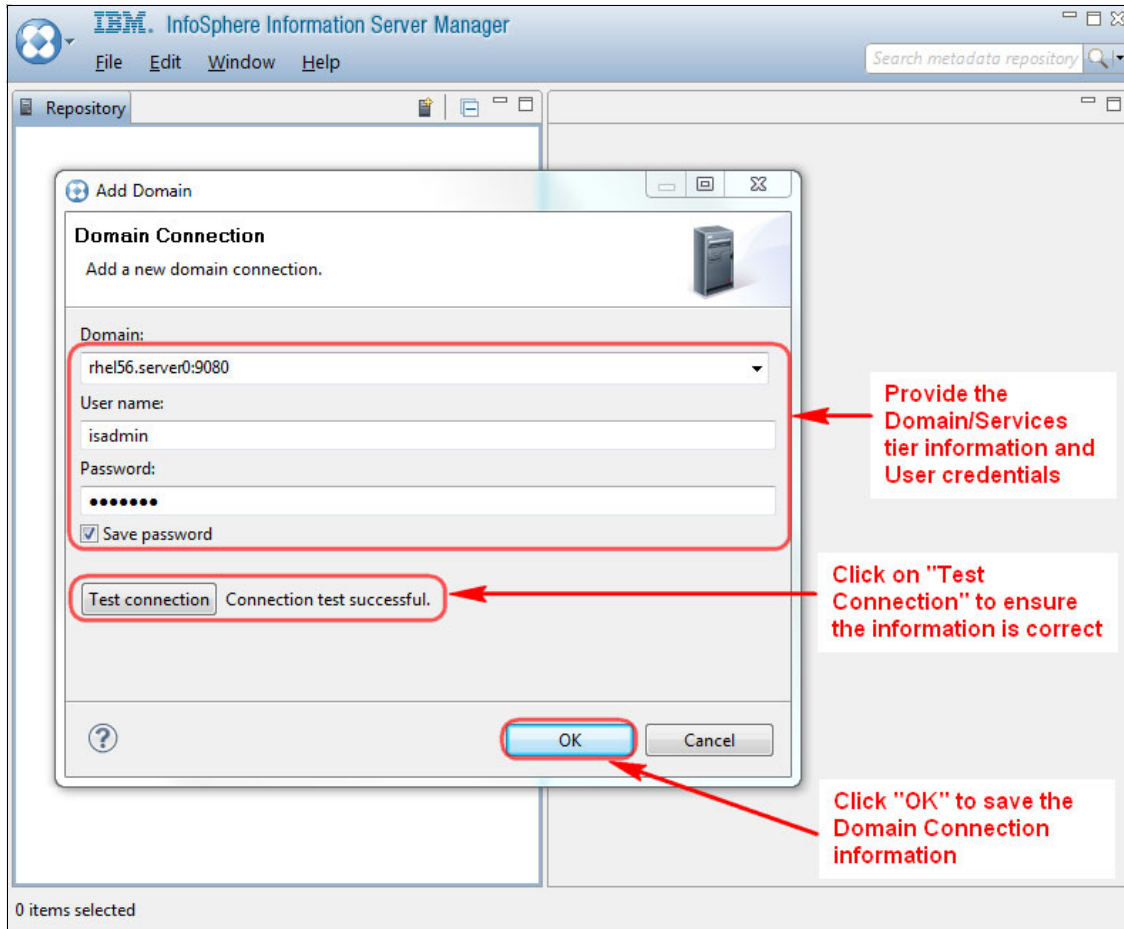


Figure 4-20 Defining an InfoSphere Information Server domain

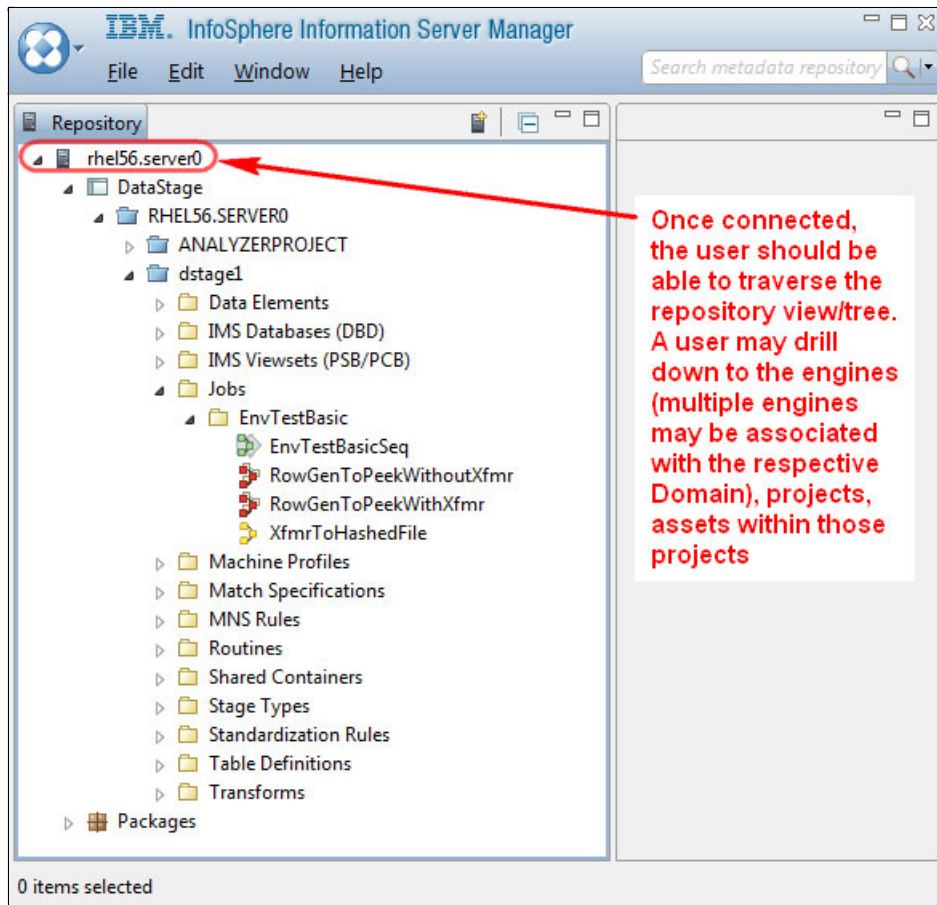


Figure 4-21 InfoSphere Information Server repository view after successful connection

4.13 Installing the IBM Rational ClearTeam Explorer Eclipse plug-in

Now that the InfoSphere Information Server Manager has the InfoSphere Information Server domain defined, the next step is to install the Eclipse plug-in of source control tool. In this case, because we are using ClearCase as the source control tool, the plug-in has either of these names:

- ▶ IBM Rational ClearTeam Explorer for Eclipse plug-in
- ▶ IBM Rational ClearCase Remote Client for Eclipse plug-in

The IBM Rational ClearTeam Explorer is used to access and modify resources under Rational ClearCase source control. The IBM Rational ClearTeam Explorer is a user interface that is powered by Eclipse technology and may be used by developers over both a local area network (LAN) or a wide area network (WAN).

The figures in this section (Figure 4-22 on page 109 through Figure 4-28 on page 114) illustrate how to start the ClearTeam Explorer for Eclipse plug-in installation process:

1. Start the installation by selecting **Help** → **Install New Software**.
2. Add a software site by clicking **Add**.
3. Provide details for the site by adding a name and location.
4. Select the features to be installed.
5. Review the details of the installation and click **Next**.
6. Review and accept the license, and then click **Next**.
7. Restart the InfoSphere Information Server Manager by clicking **Restart Now**.

Name: In the earlier versions of ClearCase, the IBM Rational ClearTeam Explorer for Eclipse plug-in is sometimes referred to as Rational ClearCase Remote Client for Eclipse Plug-in.

This process must be done on each client workstation of every user who intends to perform source control integration.

Installing plug-ins for other source code control systems have similar steps.

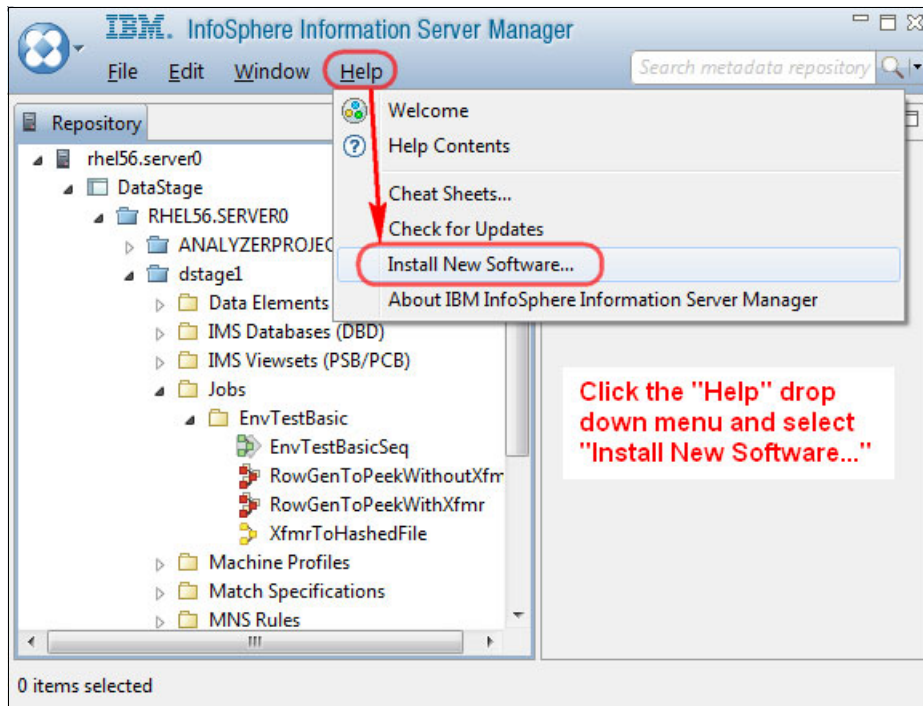


Figure 4-22 Starting the Eclipse plug-in installation process

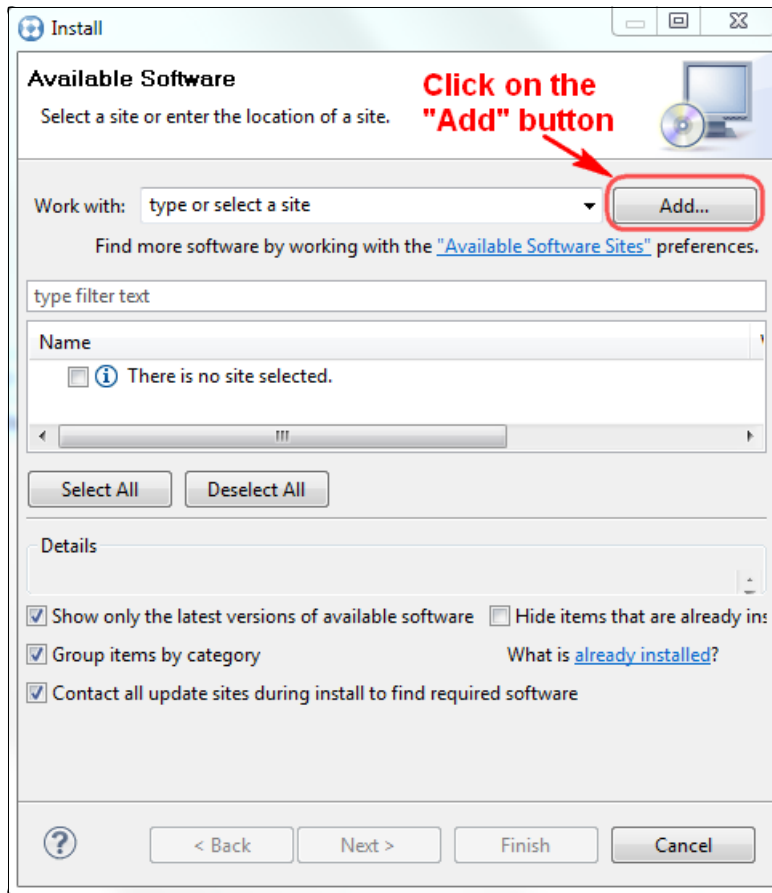


Figure 4-23 Adding a software site for the Eclipse plug-in download

In Figure 4-24 on page 111, note that the Name field in the Add Repository dialog can be any preferred name. However, use the following format in the Location field, where <HostName> is the host name of the ClearCase server:

```
http://<HostName>:9080/ccrc/update
```

The values in Figure 4-24 on page 111 are only examples. Use the host name that applies to the actual ClearCase server in your environment.

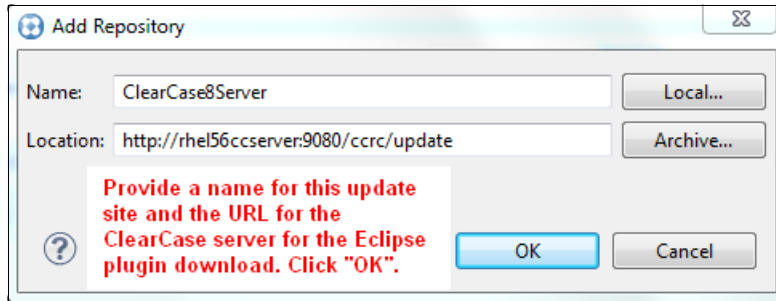


Figure 4-24 Providing details of the ClearCase Server (software site)

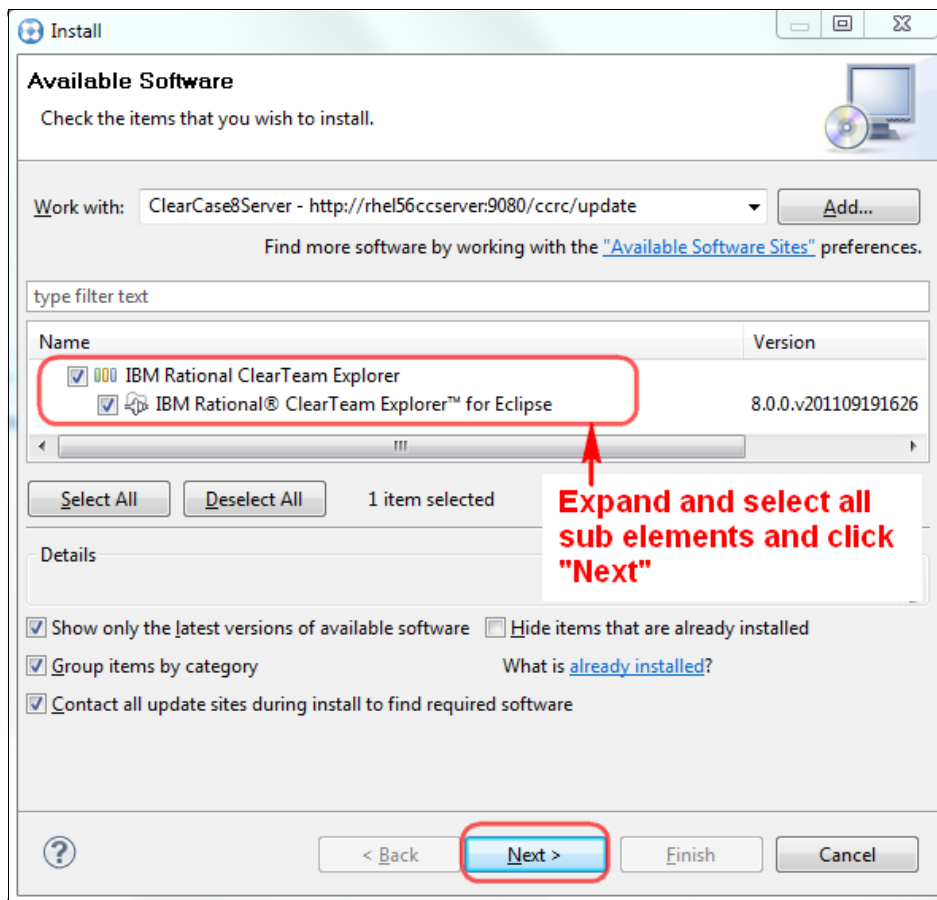


Figure 4-25 Selecting the Eclipse plug-in features to be installed

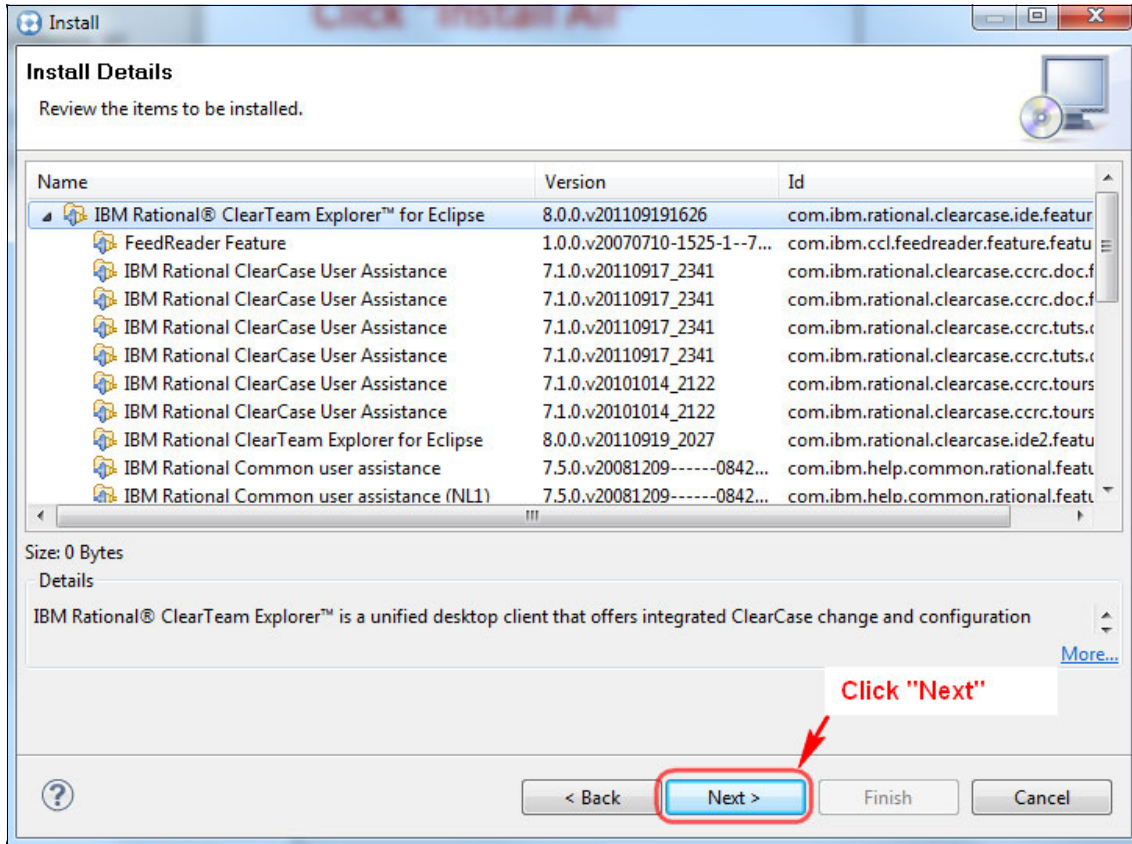


Figure 4-26 Review panel

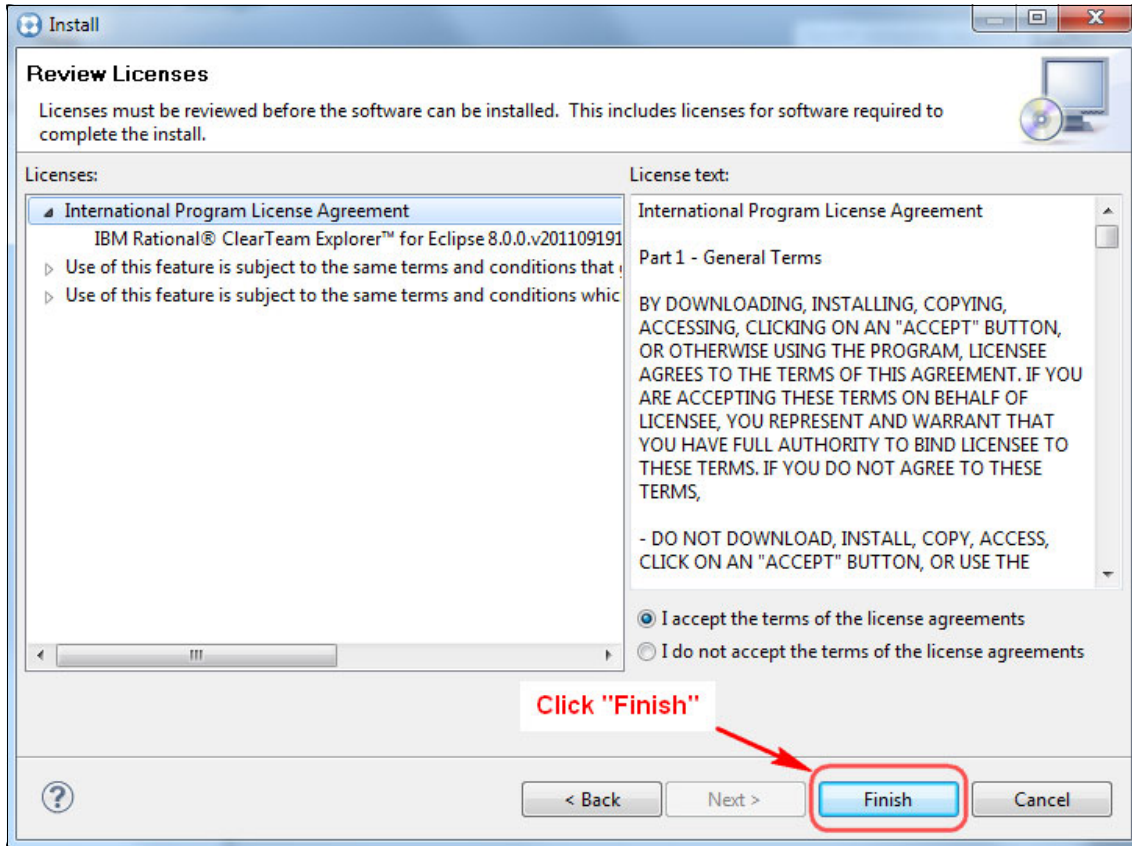


Figure 4-27 Accept license terms and click Finish

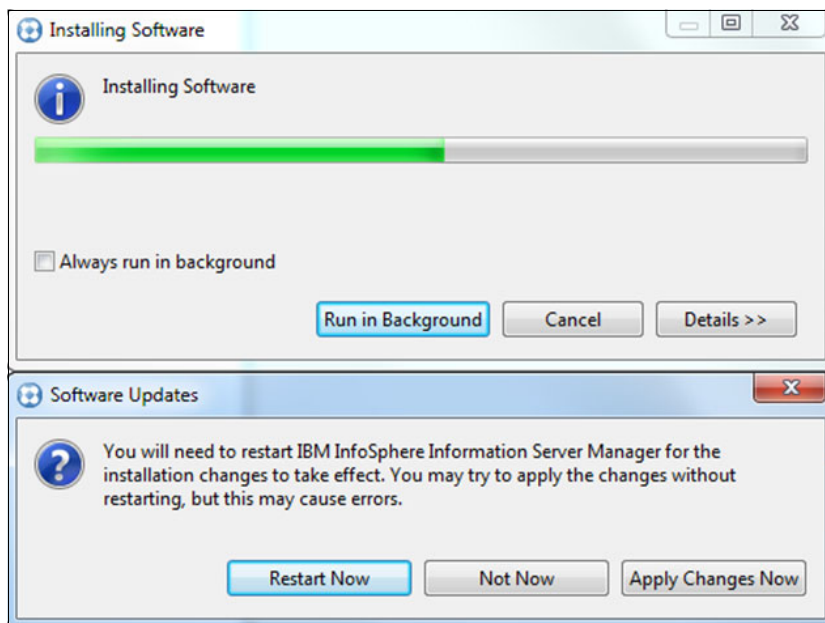


Figure 4-28 Restart InfoSphere Information Server Manager

4.14 Integrating source control

The steps in this section should be performed only once, for a given domain. The result is the creation of assets within the source control repository in the ClearCase server, and also a local workspace in the Windows machine where the administrator would be performing these steps.

Figure 4-29 on page 115 through Figure 4-35 on page 121 illustrate the procedure for integrating a source control system into InfoSphere Information Server Manager:

1. Start the integration process. by right-clicking the domain name and selecting **Integrate Source Control**. Be sure the **Share project** check box is selected and click **OK**.
2. Create a ClearCase view:
 - a. Select the plug-in for sharing the project, click **Next**, and then click **Create View**.
 - b. Select **Create a base ClearCase view**, click **Next**, provide user credentials, and click **OK**.
 - c. Click **Next**, provide the view tag name and path, and then click **Finish**.

3. Select a location for the project folder.
4. Click **OK**.
5. Browse the new source control workspace.

Other developers and release managers must import the source control workspace, which is described in 4.15, “Importing a source control project” on page 122.

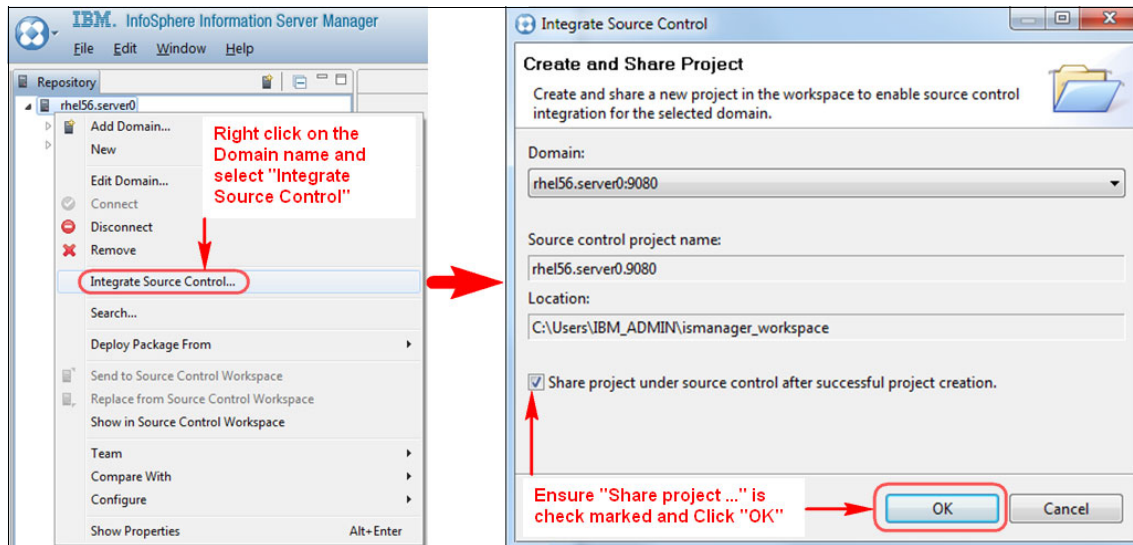


Figure 4-29 Starting the source control integration process

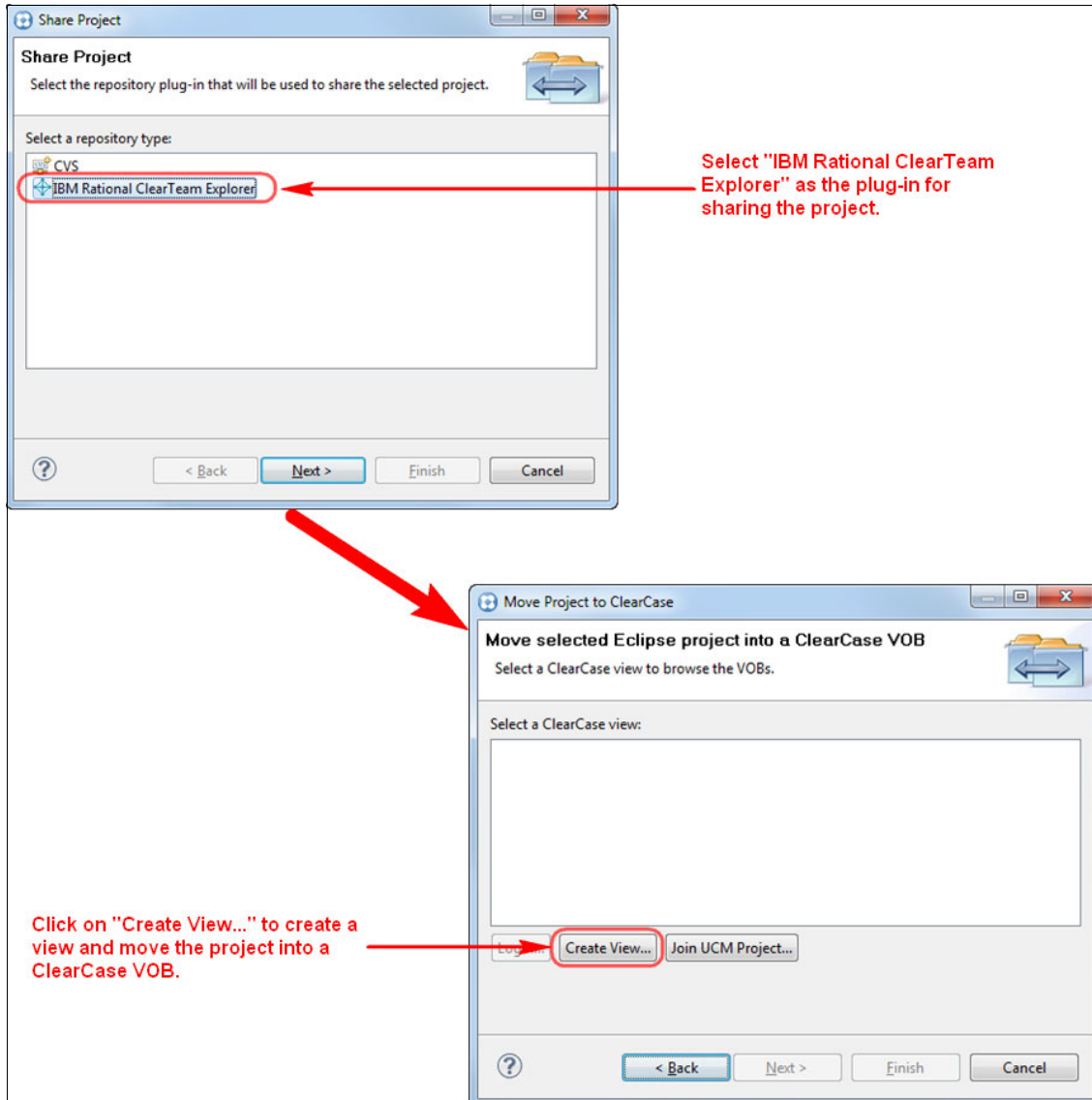


Figure 4-30 Creating a ClearCase view (Part 1 of 3)

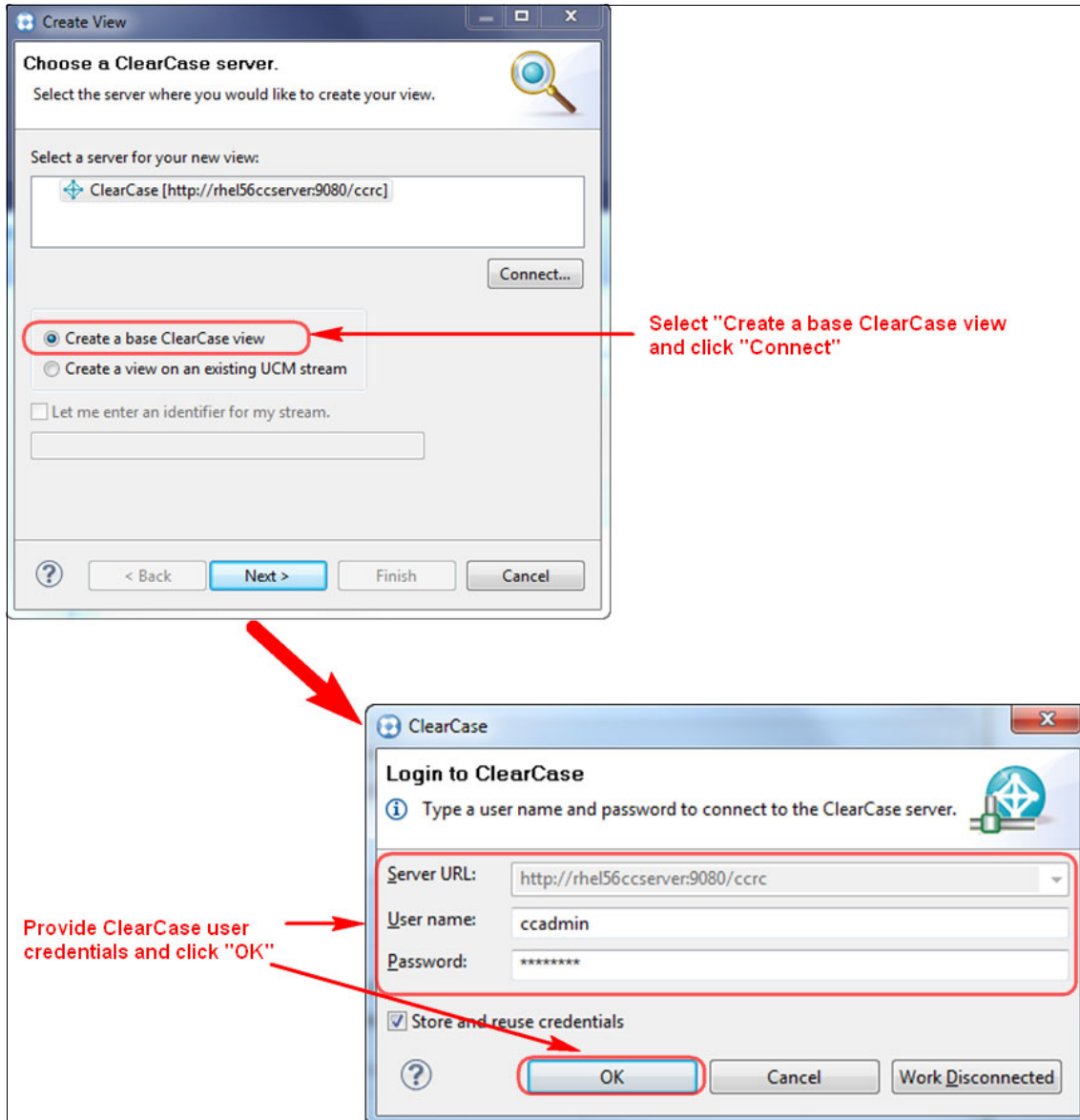


Figure 4-31 Creating a ClearCase view (Part 2 of 3)

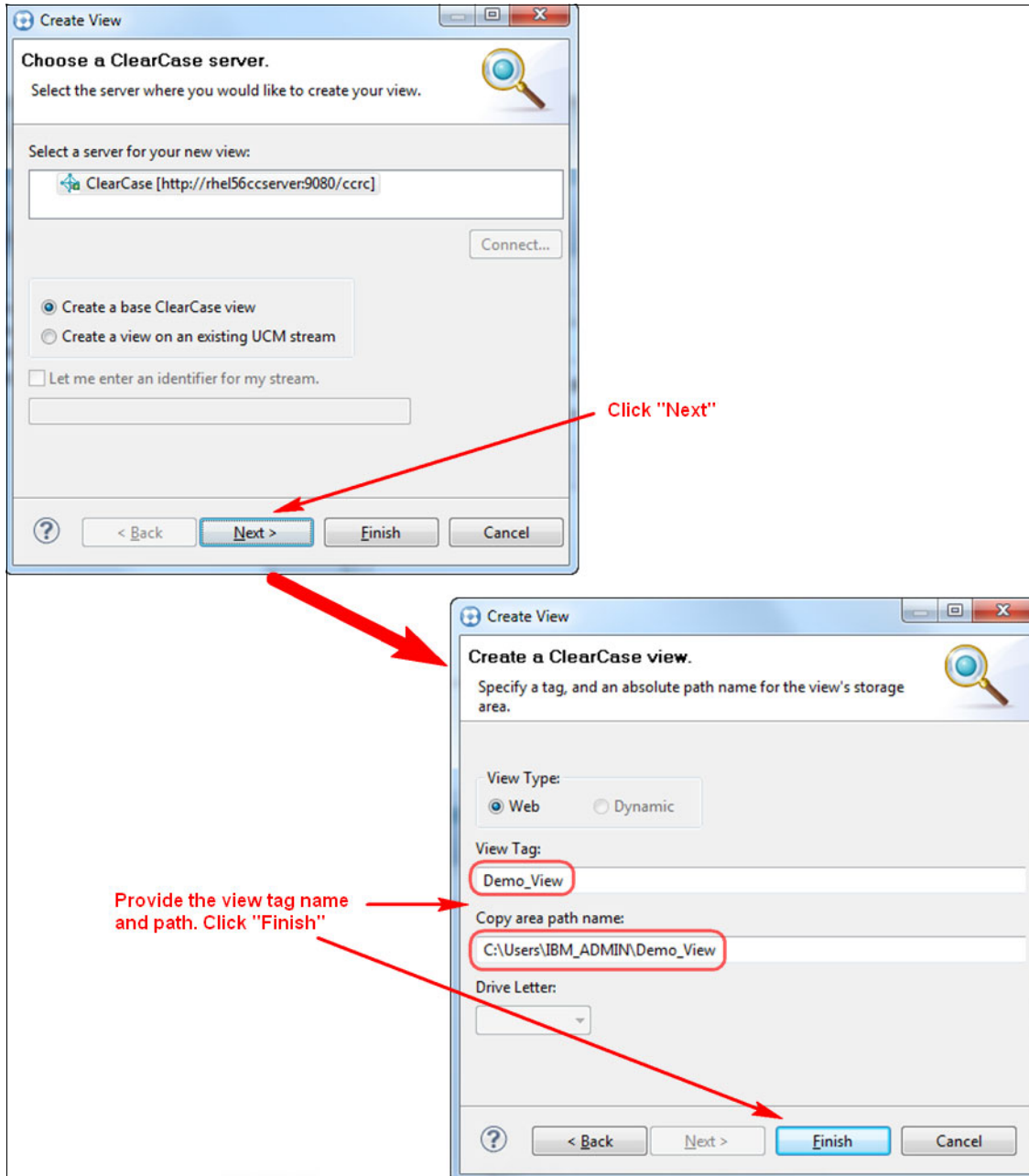


Figure 4-32 Creating a ClearCase view (Part 3 of 3)

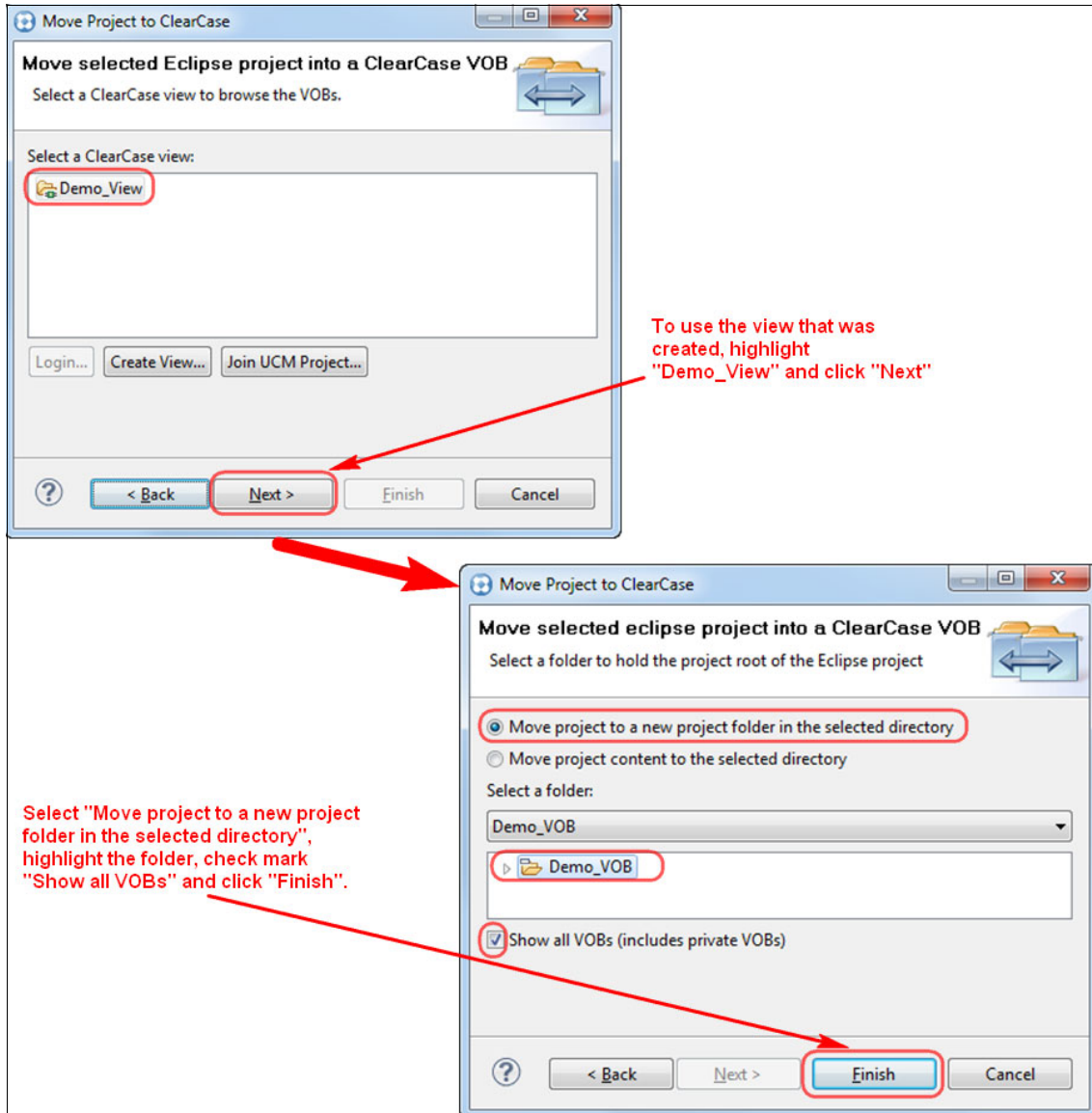


Figure 4-33 Selecting a location for the project folder

Figure 4-34 shows how to finish the setup.

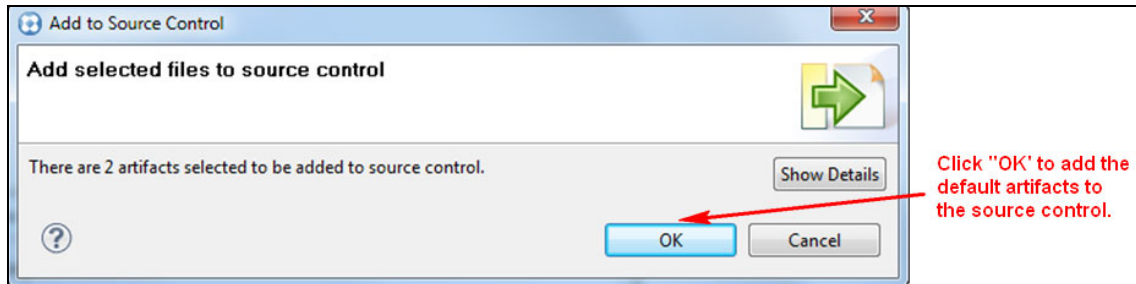


Figure 4-34 Finishing the setup of source control integration process

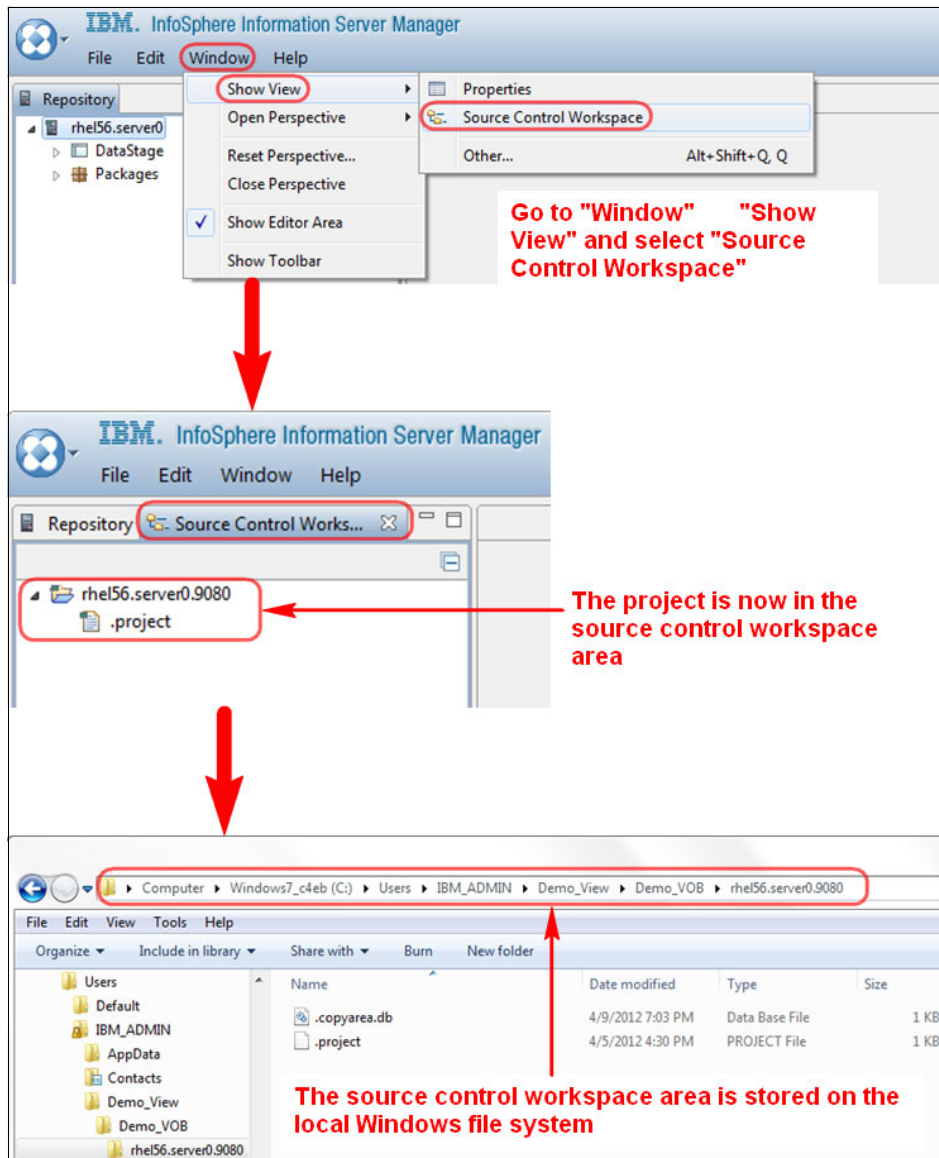


Figure 4-35 Browsing the newly created source control workspace in InfoSphere Information Server Manager and on the local Windows file system

4.15 Importing a source control project

All other developers and release managers must import the source control project that was created in 4.14, “Integrating source control” on page 114. The result is a local area in the Windows client workstation for each user. The process pulls the files and directories from the ClearCase repository into the local workspace directory.

Be sure to first complete the procedure in 4.14, “Integrating source control” on page 114. The figures (Figure 4-36 through Figure 4-42 on page 128) in this section illustrate the procedure for importing an existing source control project for additional users:

1. Select **Window** → **Show View** → **Source Control Workspace** to see the workspace.
2. Select **Window** → **Open Perspective** → **Other**. In the Open Perspective dialog, make a selection and click **OK**.
3. Right-click **My Views** and then select **Connect**. Enter the connection information details, and click **OK**.
4. Create a view by right-clicking **My Views**, selecting **Create View**, selecting **Create a base ClearCase view**, and then clicking **Next**.
5. Select resources for the view.
6. Import the InfoSphere Information Server domain folder into the local workspace.
7. Browse the imported project in the newly created workspace.

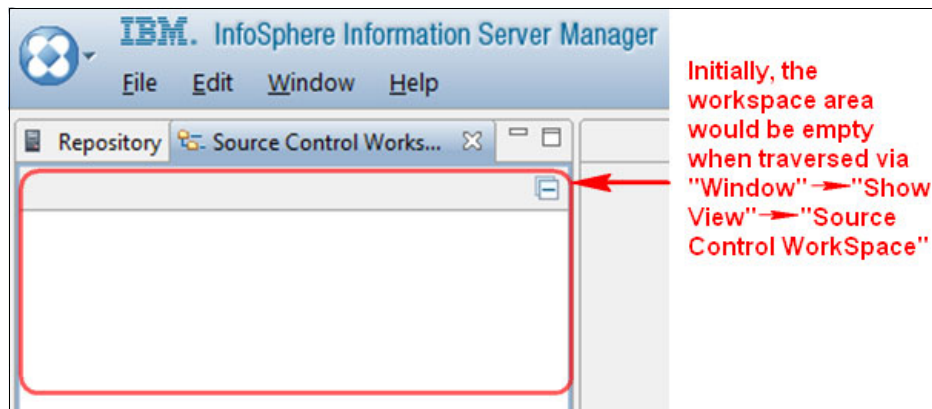


Figure 4-36 The Source Control Workspace area is empty for first-time additional users

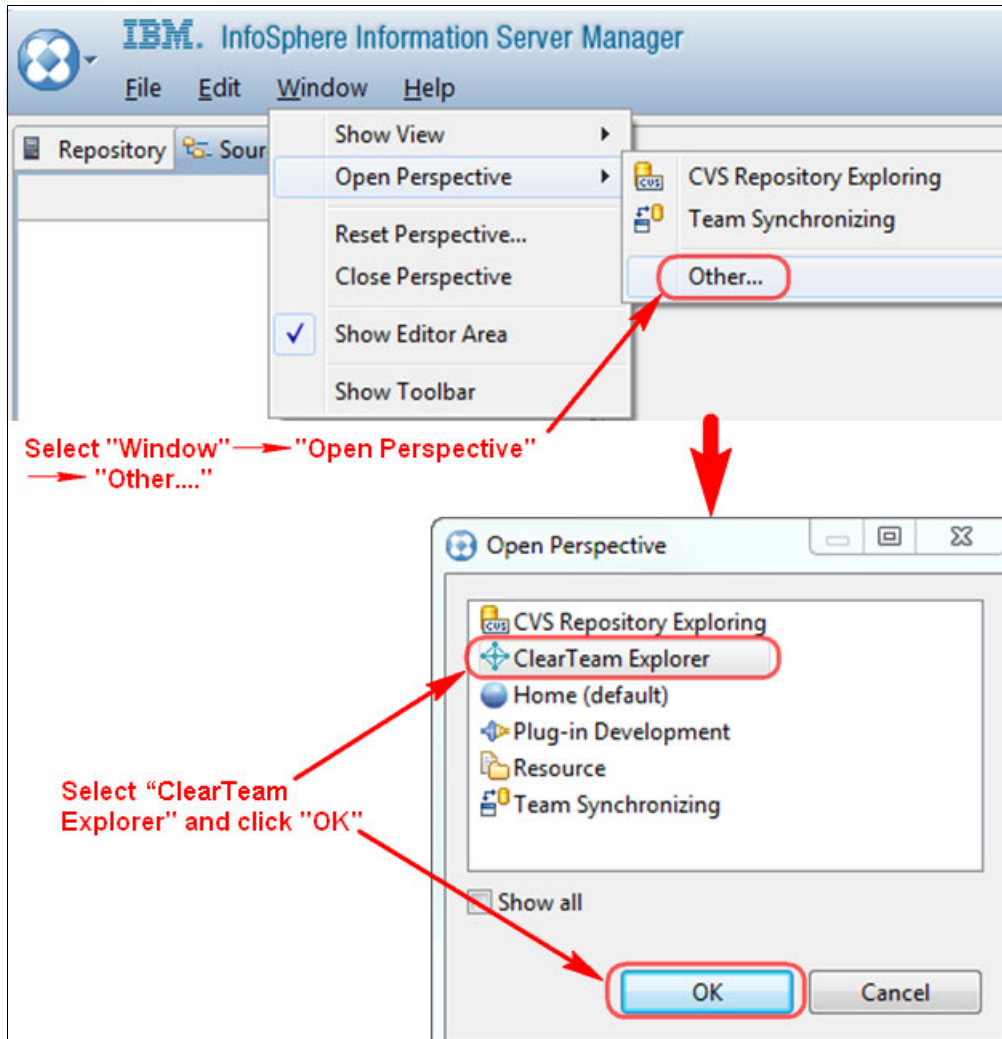


Figure 4-37 First steps for additional first-time users

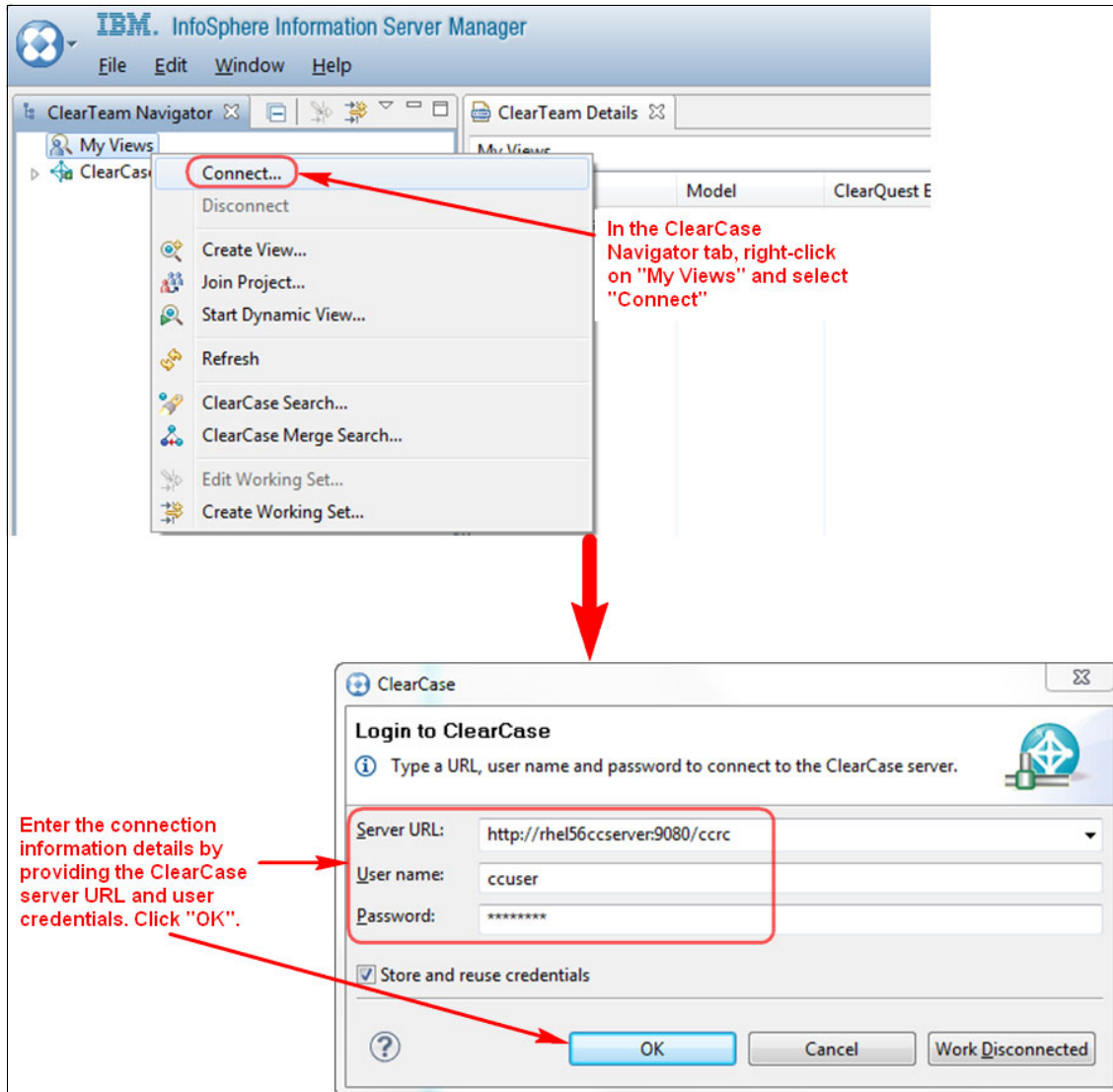


Figure 4-38 Connecting to the ClearCase server from within ClearTeam Navigator

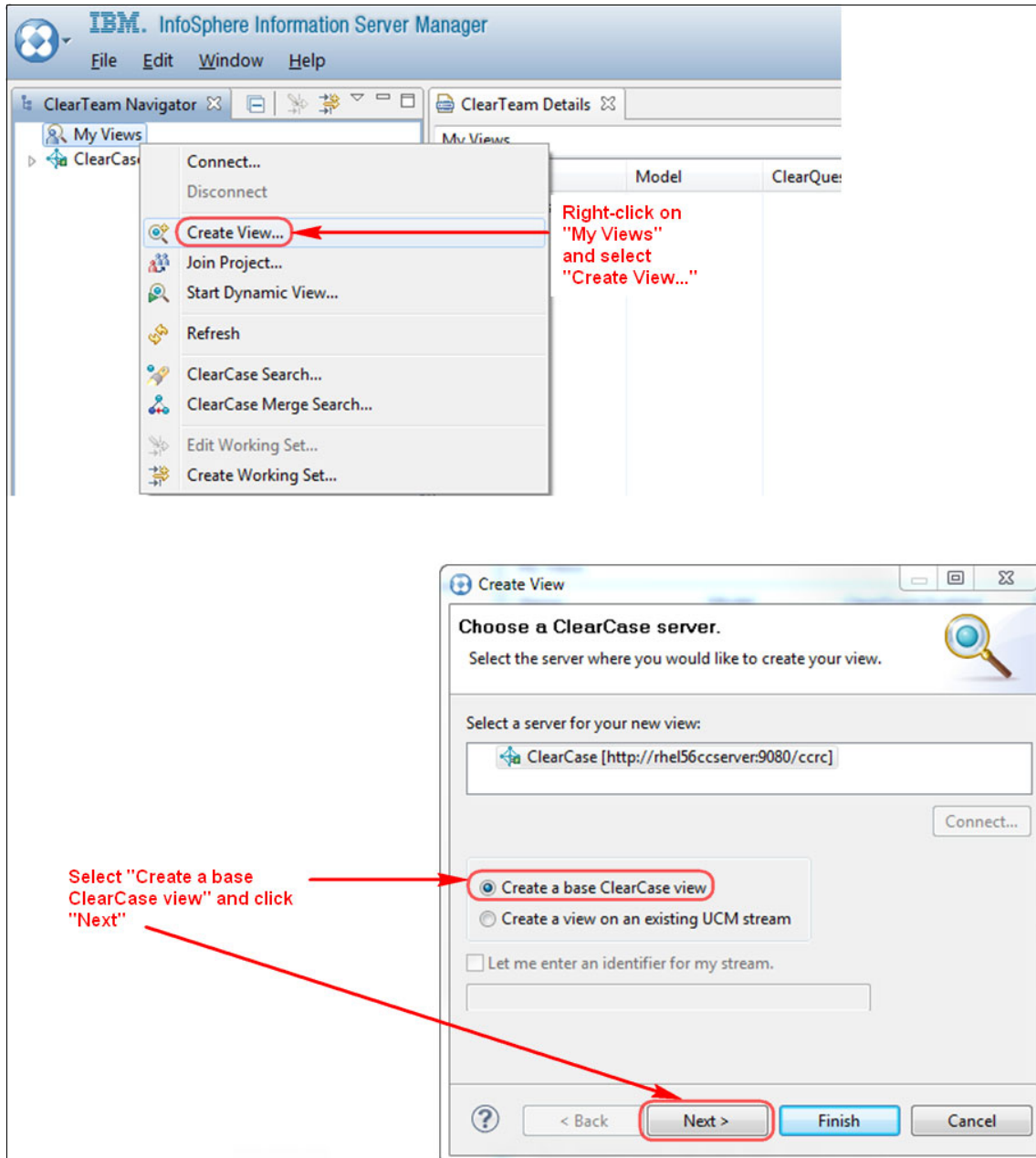


Figure 4-39 Creating a base ClearCase view by using ClearTeam Navigator

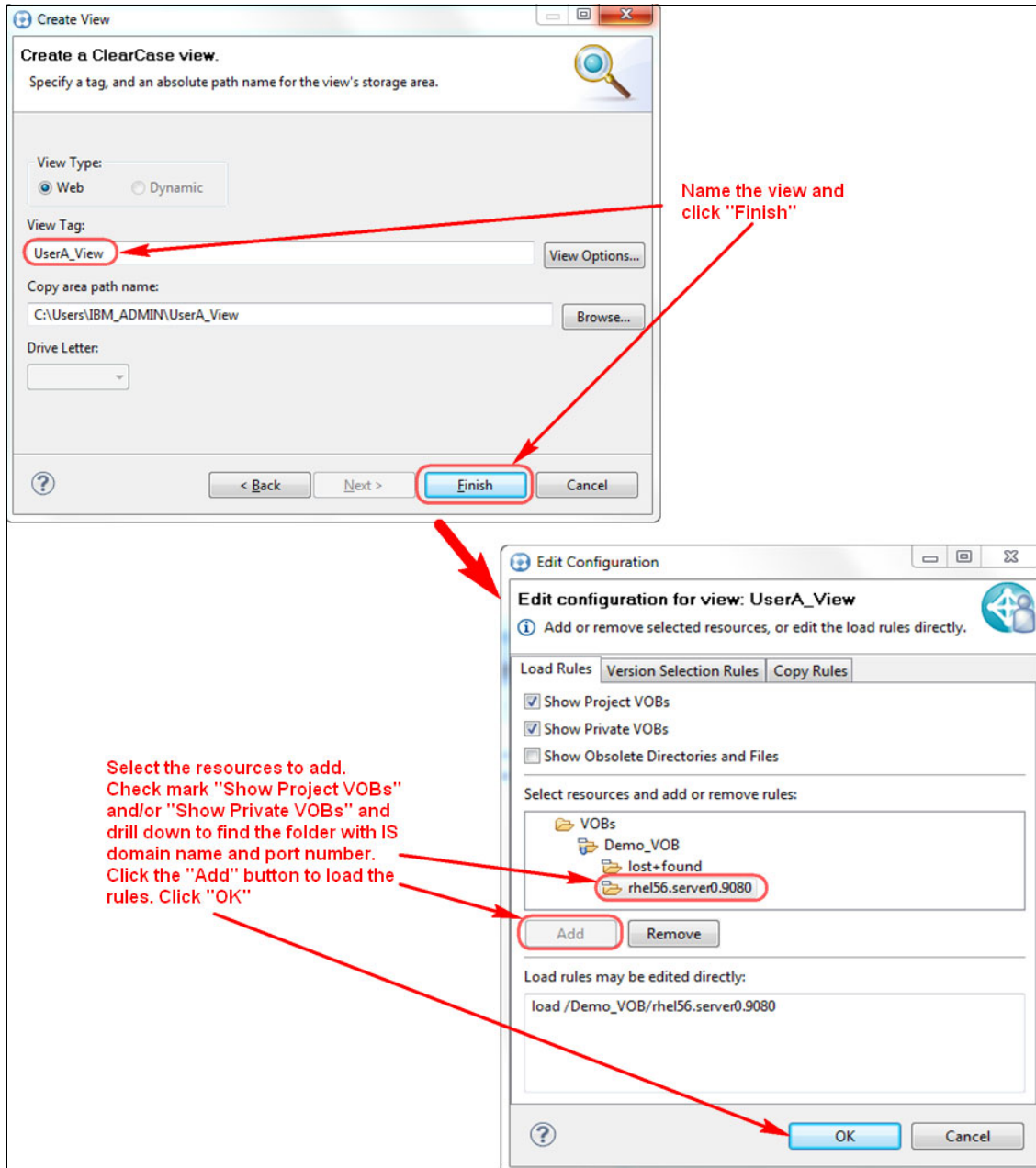


Figure 4-40 Selecting resources for the view

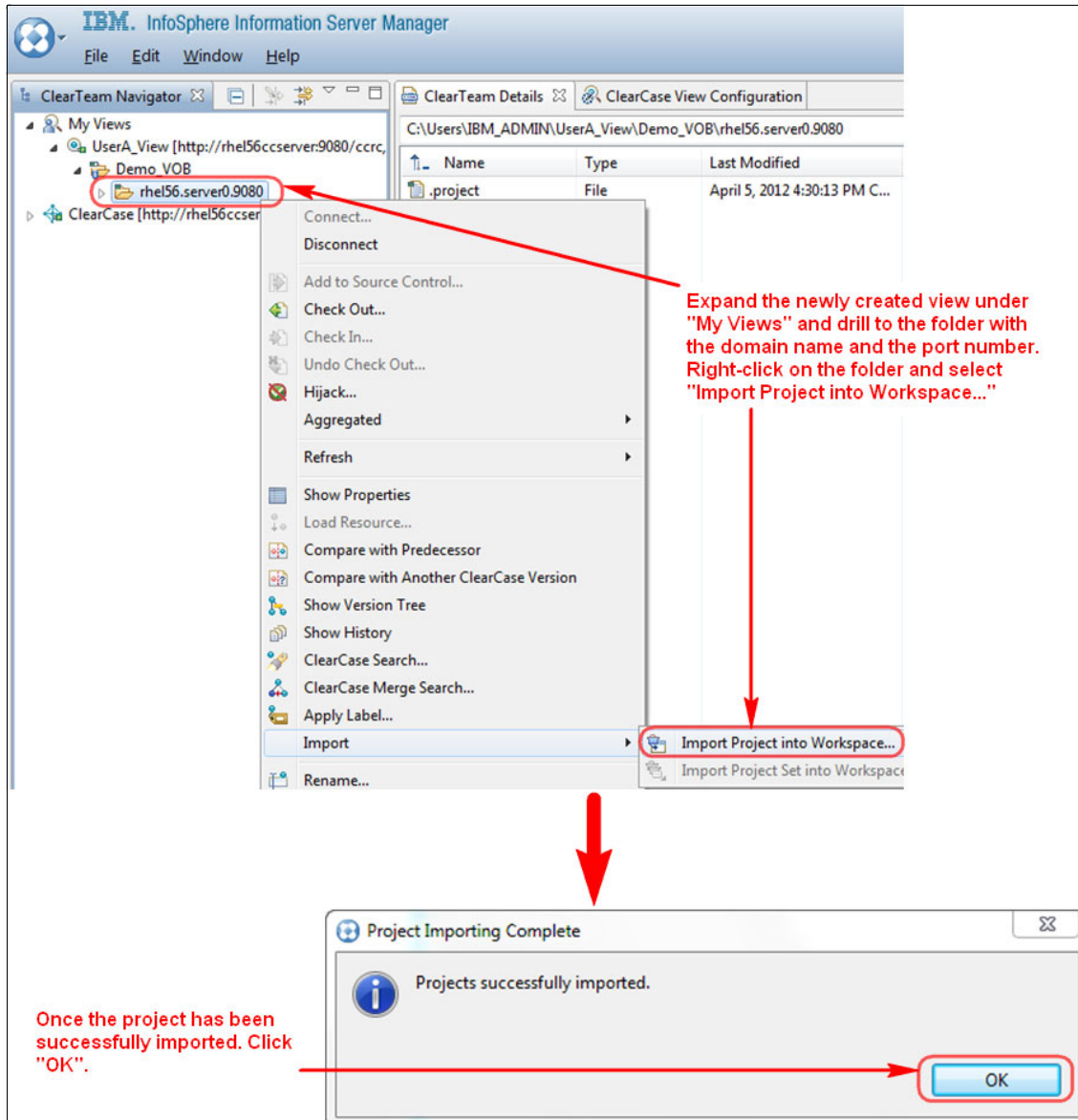


Figure 4-41 Importing the InfoSphere Information Server domain folder into the local workspace

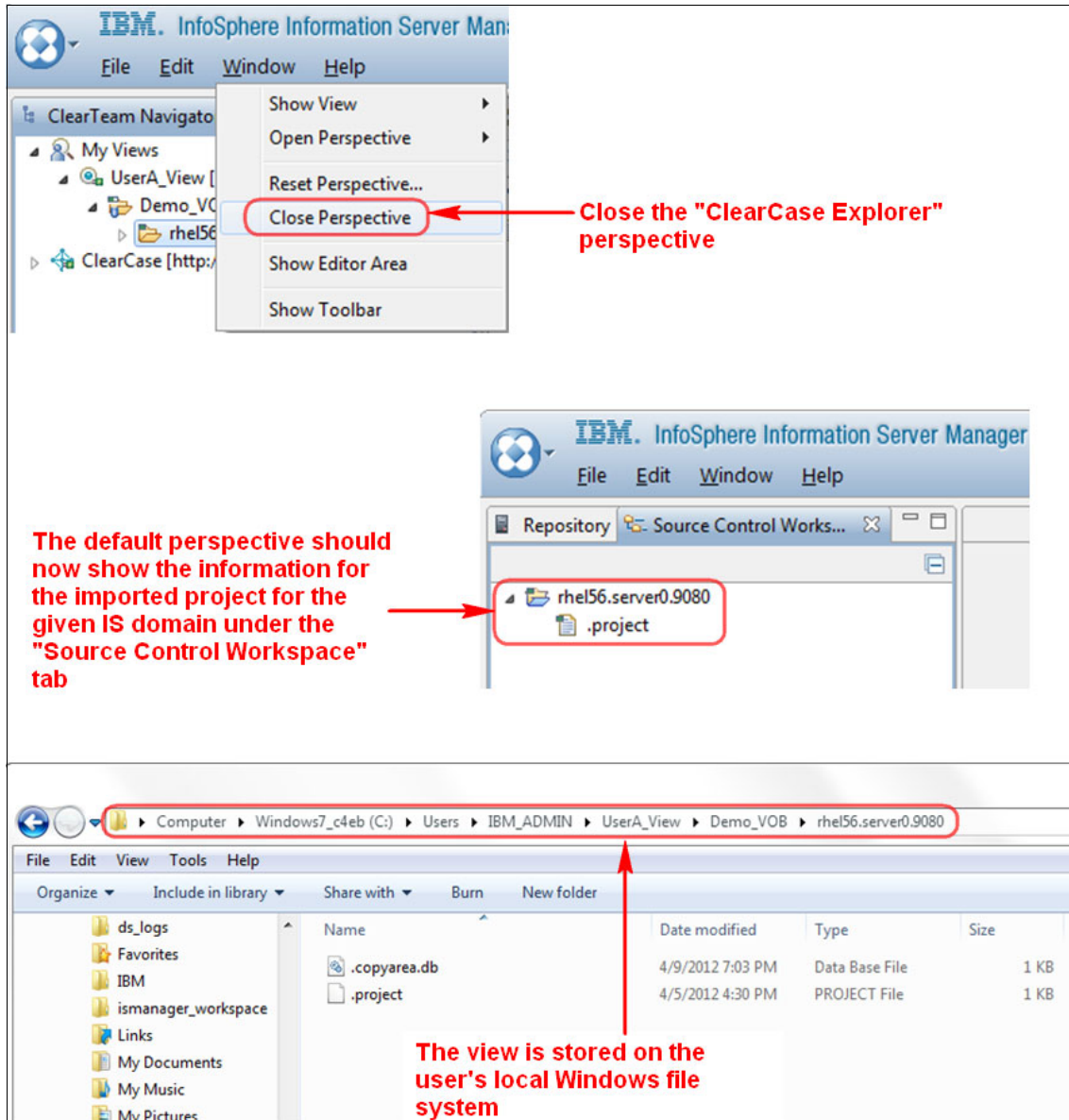


Figure 4-42 Browsing the imported project in the newly created workspace

4.16 Adding assets to Source Control Workspace

This section describes how to add a new DataStage asset (such as a job or a job sequence) from the InfoSphere Information Server repository to the Source Control Workspace.

By adding to the Source Control Workspace, the asset is also uploaded to the ClearCase repository implicitly. See Figure 4-43 on page 130.

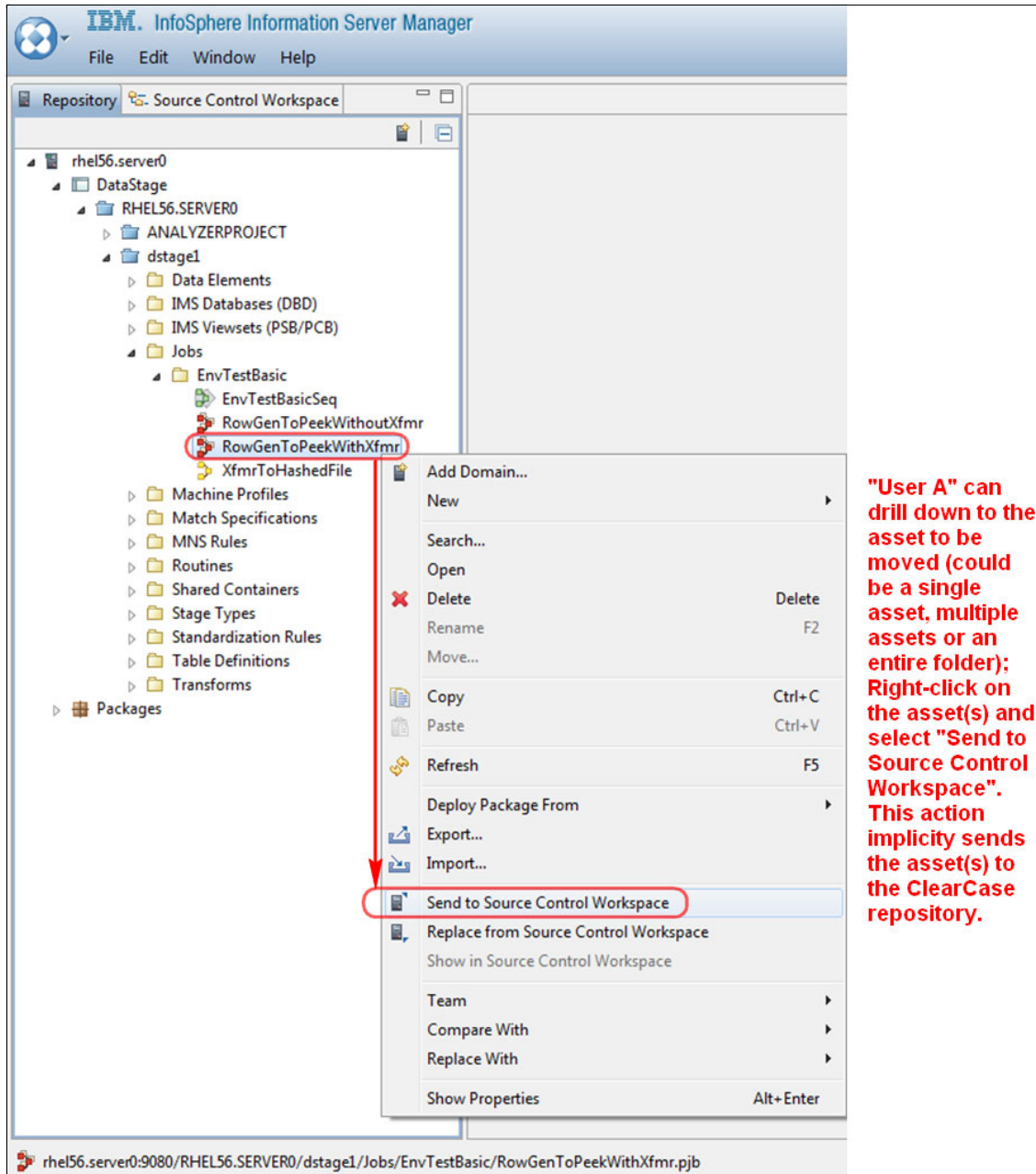


Figure 4-43 Moving assets to the Source Control Workspace

Figure 4-44 highlights the label/icon decorators. Most source control systems and tools have their own version of label/icon decorators. For example, CVS has a different depiction of label/icon decorators, and so on. Because ClearCase is the source control system being used in this publication, the ClearCase label/icon decorators are used.

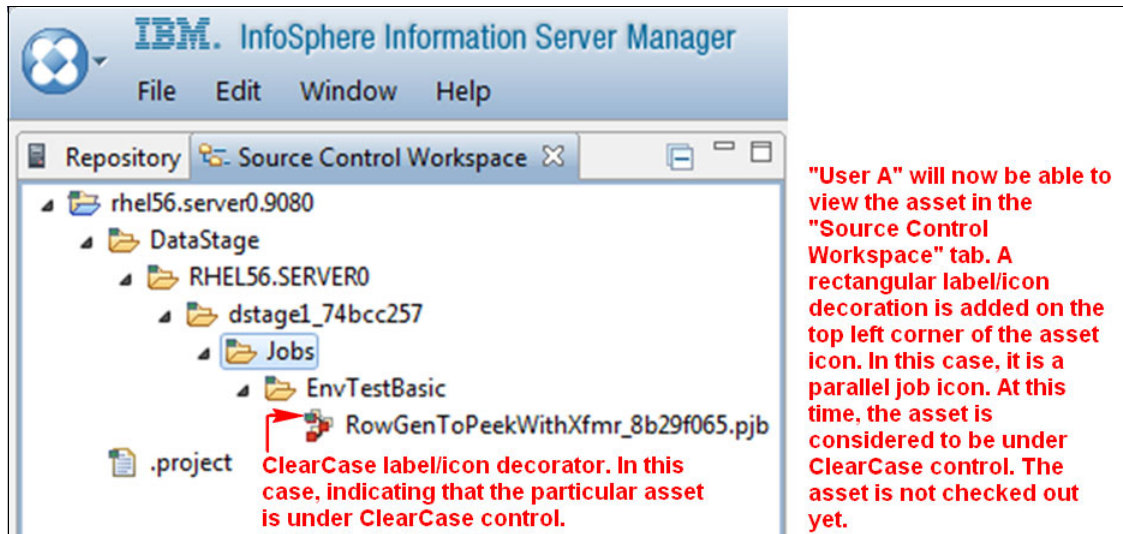


Figure 4-44 Viewing assets in the Source Control Workspace

In ClearCase terminology, *label decorators* indicate the state of a resource in a ClearCase view and allow additional information to be displayed in the label and icon of an asset. Figure 4-45 and Figure 4-46 on page 133 show the label decorators that are used by the ClearCase version that is used this publication. The label/icon decorators can differ from one version to another of a particular source control system.








ClearCase state	Icon decoration	Text decoration	Resource type	Description
Under ClearCase control		@@	File or directory	The resource is under ClearCase control and has been loaded into the view's copy area.
View private		none	File or directory	The resource is an ordinary file or directory in the local file system. It is not under ClearCase control.
Partially loaded		@@PL	directory	The directory is loaded but contains one or more resources that have not been loaded into the view's copy area.
Not loaded		@@NL	File or directory	The resource is under ClearCase control but has not been loaded because it is not selected by the ClearCase view's load rules.
Upgrade view		none	View	The view was created in a version of ClearCase prior to 7.1. The view needs to be upgraded if it is to be used in 7.1. Note that if you upgrade the view to 7.1, it will no longer be functional or visible in prior versions of ClearCase.
Checked out		@@CO	File or directory	The resource is checked out.
Aggregated change		none	File or directory	At some point at or below the current node level, a resource has been changed (checked out or hijacked).

Figure 4-45 ClearCase label/icon decorators (Part 1 of 2)














Aggregated and checked out		none	Directory	An aggregated directory resource is checked out and may also have checkouts.
Hijacked		@@HJ	File	The file is hijacked.
Unknown		@@?	File or directory	The resource has an unknown ClearCase state.
VOB		none	directory	The resource is a ClearCase VOB.
UCM resource		none	View, file, or directory	The resource is part of a ClearCase UCM project.
Discordant state		none	File or directory	The resource is in a discordant state.
Symbolic link		none	File or directory	The resource is a symbolic link.
Component VOB		none	VOB	The resource is a Rational ClearCase component VOB.
Project VOB		none	VOB	The resource is Rational ClearCase project VOB.
ClearQuest project		none	ClearCase project	The resource is a Rational ClearCase UCM project that is integrated with Rational ClearQuest.
Obsolete		none	Stream, project, VOB, activity	The resource has been designated as Obsolete through the ClearCase Properties view.
Locked		none	Stream, project, VOB, activity	The resource has been designated as Locked through the ClearCase Properties view.
Current activity		none	Activity	The resource is the current activity.

Figure 4-46 ClearCase label/icon decorators (Part 2 of 2)

Figure 4-47 illustrates the procedure for accessing the ClearCase commands through the Team menu. Any source control system commands are available through the Team menu.

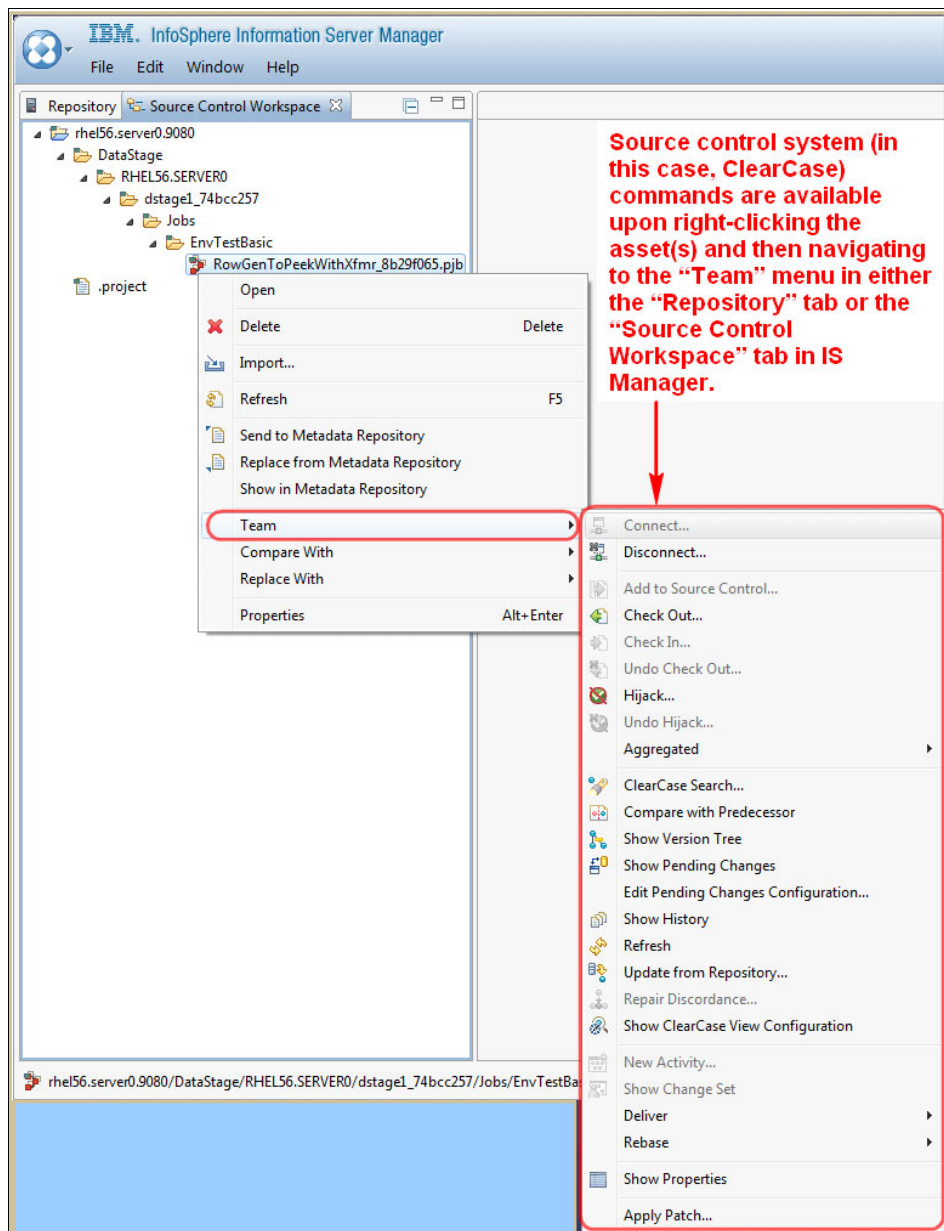


Figure 4-47 Accessing ClearCase commands through the Team menu

Figure 4-48 illustrates the procedure for viewing the version tree of an asset, as in the ClearCase repository through the Team menu.

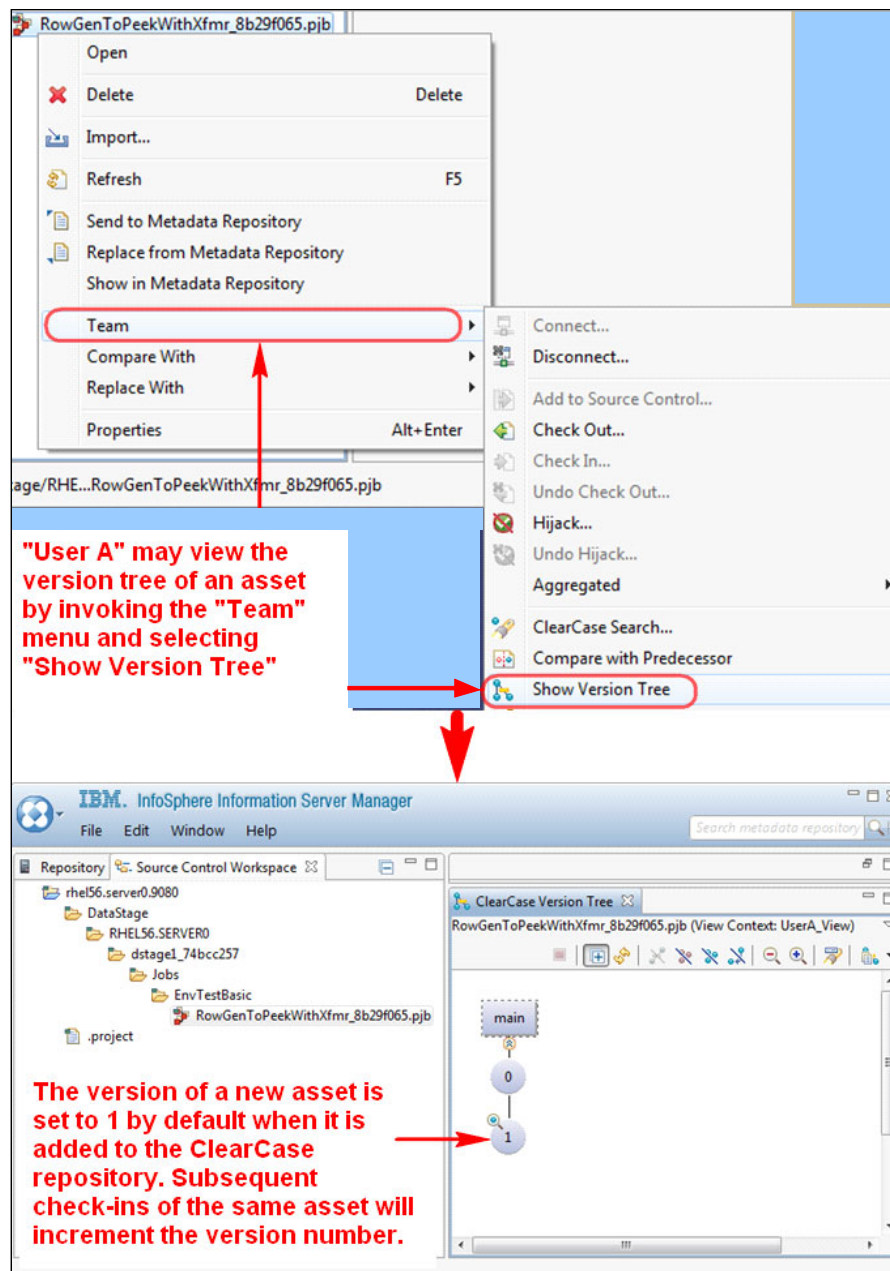


Figure 4-48 Viewing the version tree of an asset

4.17 Checking out

This section describes the steps to check out an asset that was already added to the source control repository.

Figure 4-49 on page 137 presents the check-out procedure, including an example of how the label/icon decorator of the asset changes after check-out.

The check-out option pulls the asset from the source control repository in the ClearCase server to the local source control workspace. It does not update the InfoSphere Information Server repository implicitly.

The figures in this section (Figure 4-50 on page 138 through Figure 4-53 on page 140) show two ways of updating the InfoSphere Information Server repository with the asset that was just checked out. You can use either the Repository or the Workspace view.

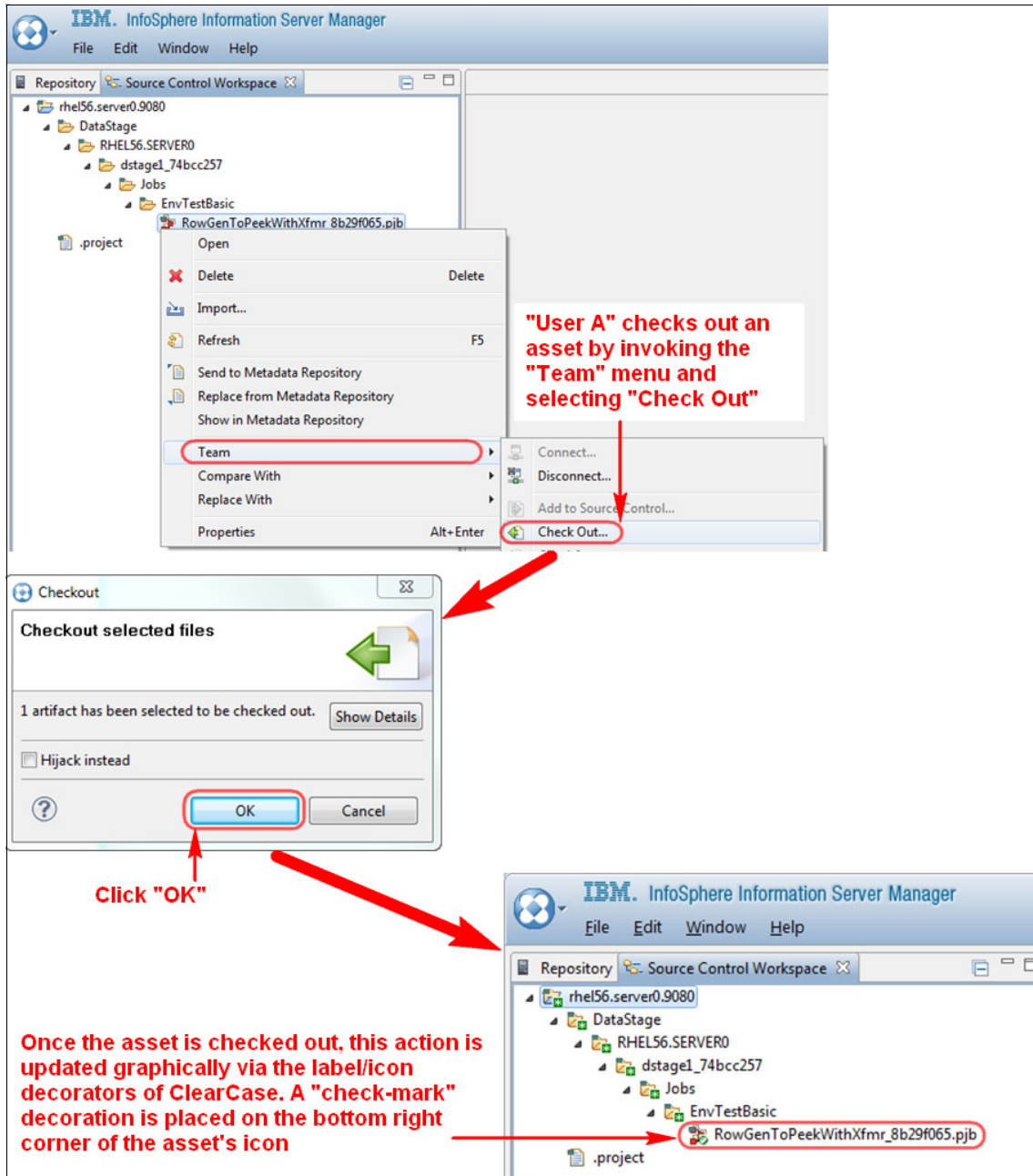


Figure 4-49 Checking out an asset using InfoSphere Information Server Manager Workspace view, through the Team menu

The user can specify additional options while checking out an asset by clicking **Show Details** in the Checkout selected files dialog (the intermediate step in Figure 4-49 on page 137) during the check-out process, as shown in Figure 4-50. A standard practice is to provide comments during checking-out and checking-in of assets from a source control system.

In ClearCase, the user can select a check-out type, as highlighted in Figure 4-50. The selected check-out type has its effect only on the assets in the ClearCase repository, and has no bearing on the assets in the InfoSphere Information Server repository.

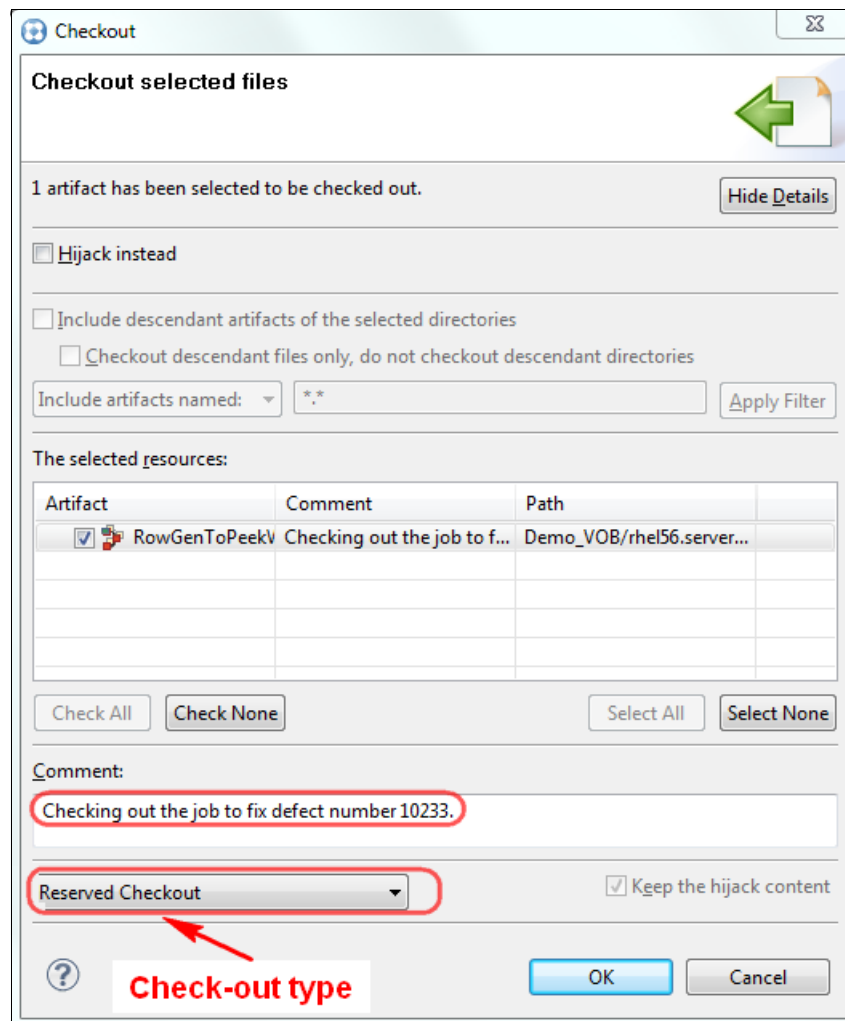


Figure 4-50 Specifying additional options during check-out process

Figure 4-51 and Figure 4-52 on page 140 illustrate the procedure for updating the xmeta repository after the assets are checked out. Either method can be used.

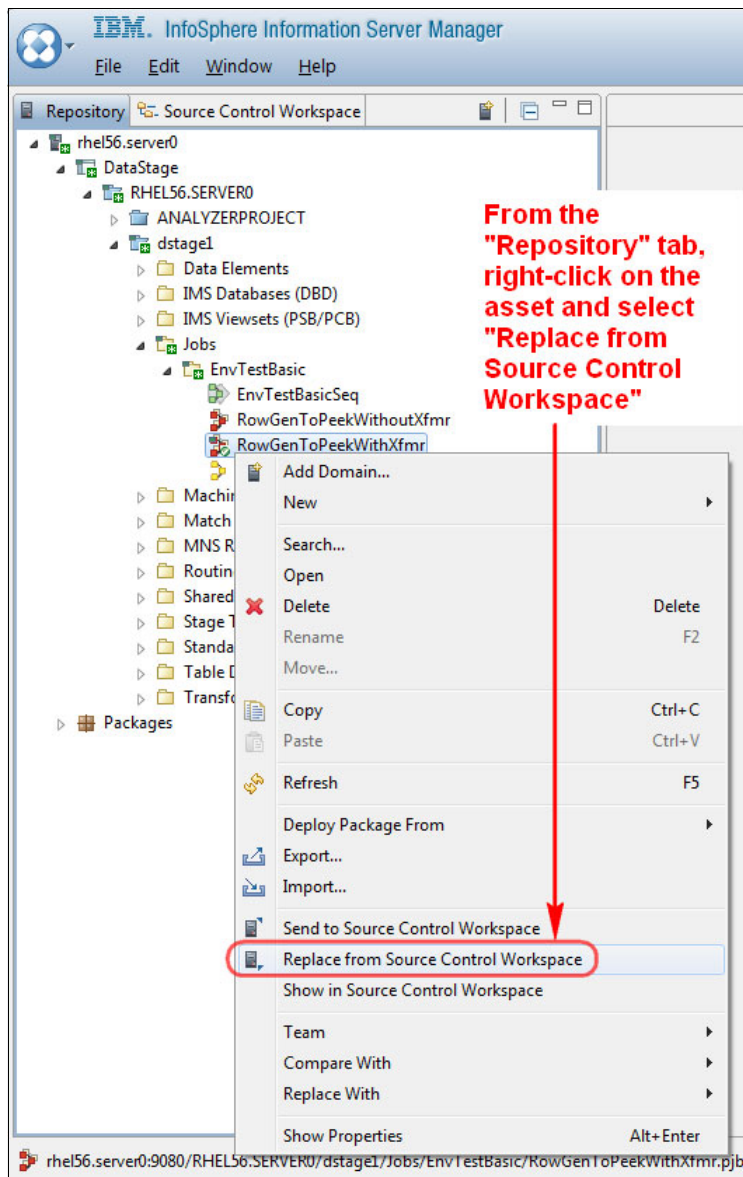


Figure 4-51 Updating xmeta after a check-out (Repository view)

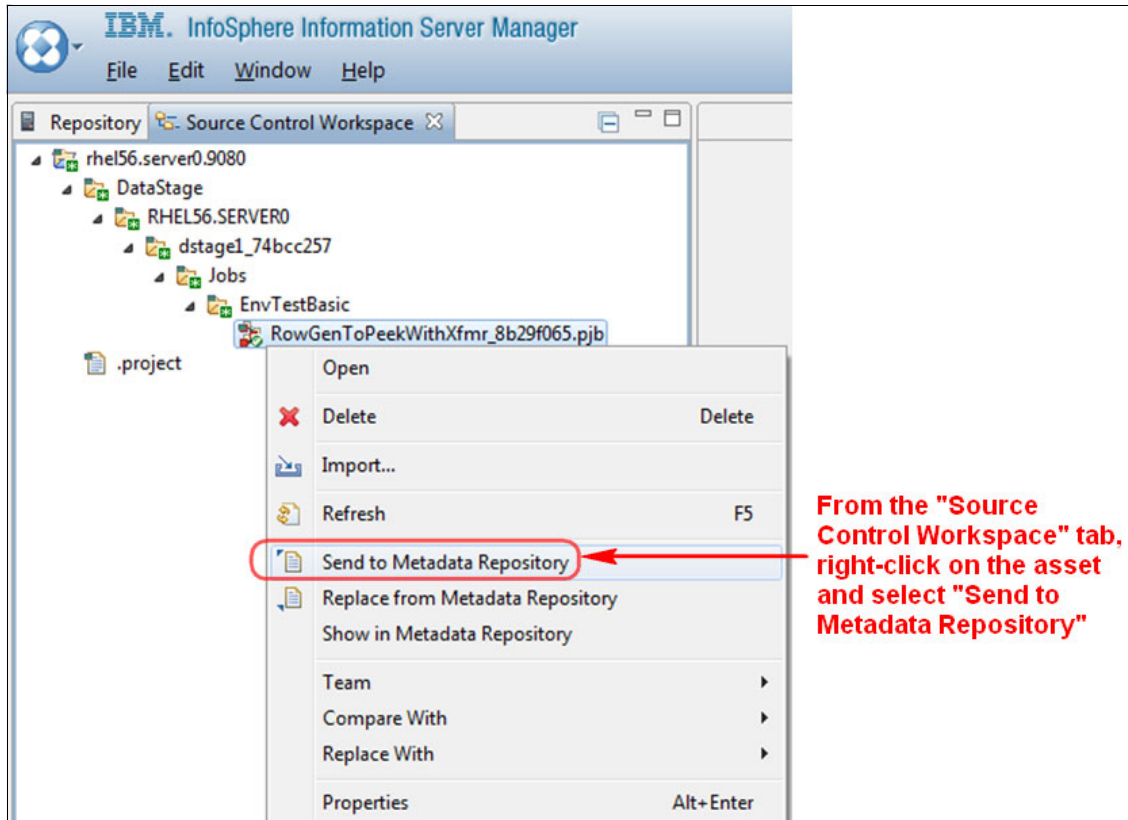


Figure 4-52 Updating xmeta after a check-out (Workspace view)

View the status of the checked-out asset, as shown in Figure 4-53.

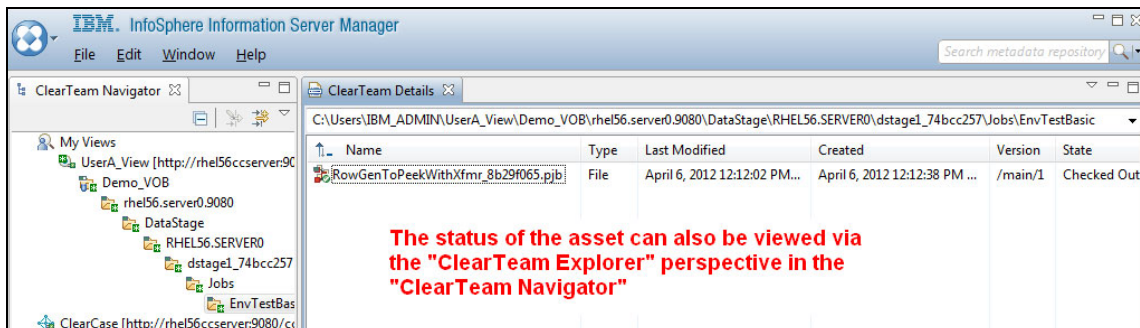


Figure 4-53 Viewing the status of a checked out asset in the ClearTeam Explorer perspective

4.18 Checking in

This section describes the steps to check an asset back into ClearCase.

The check-in option itself does not copy the DataStage asset from the InfoSphere Information Server repository into ClearCase. The check-in option moves the version that is present in the local source control workspace.

We first describe the process of updating the local source control workspace with the asset out of the InfoSphere Information Server repository. You can use either the Repository or the Source Control Workspace view. This case is typical if the asset is updated in DataStage Designer after checking out.

Figure 4-54 on page 142 and Figure 4-55 on page 143 illustrate the procedure of updating the Source Control Workspace before checking in the assets. Either method can be used.

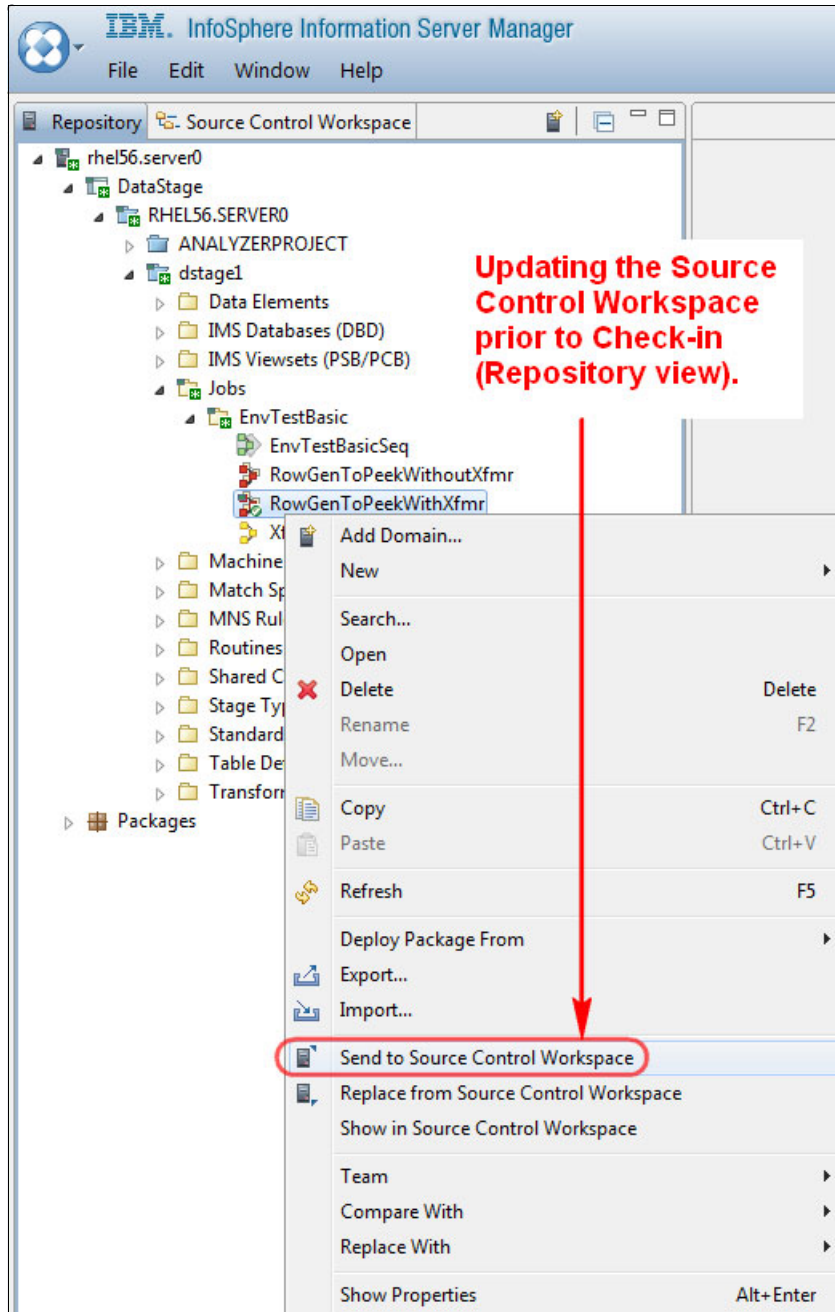


Figure 4-54 Updating the Source Control Workspace before check-in (Repository view)

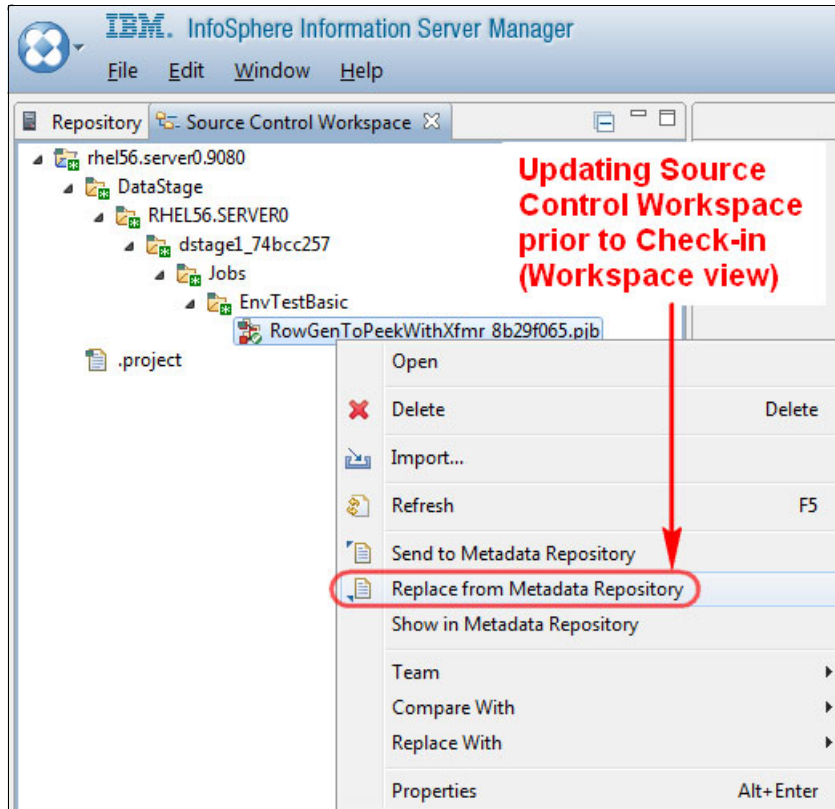


Figure 4-55 Updating the Source Control Workspace prior to check-in (Workspace view)

Figure 4-56 illustrates how to do the actual check-in process.

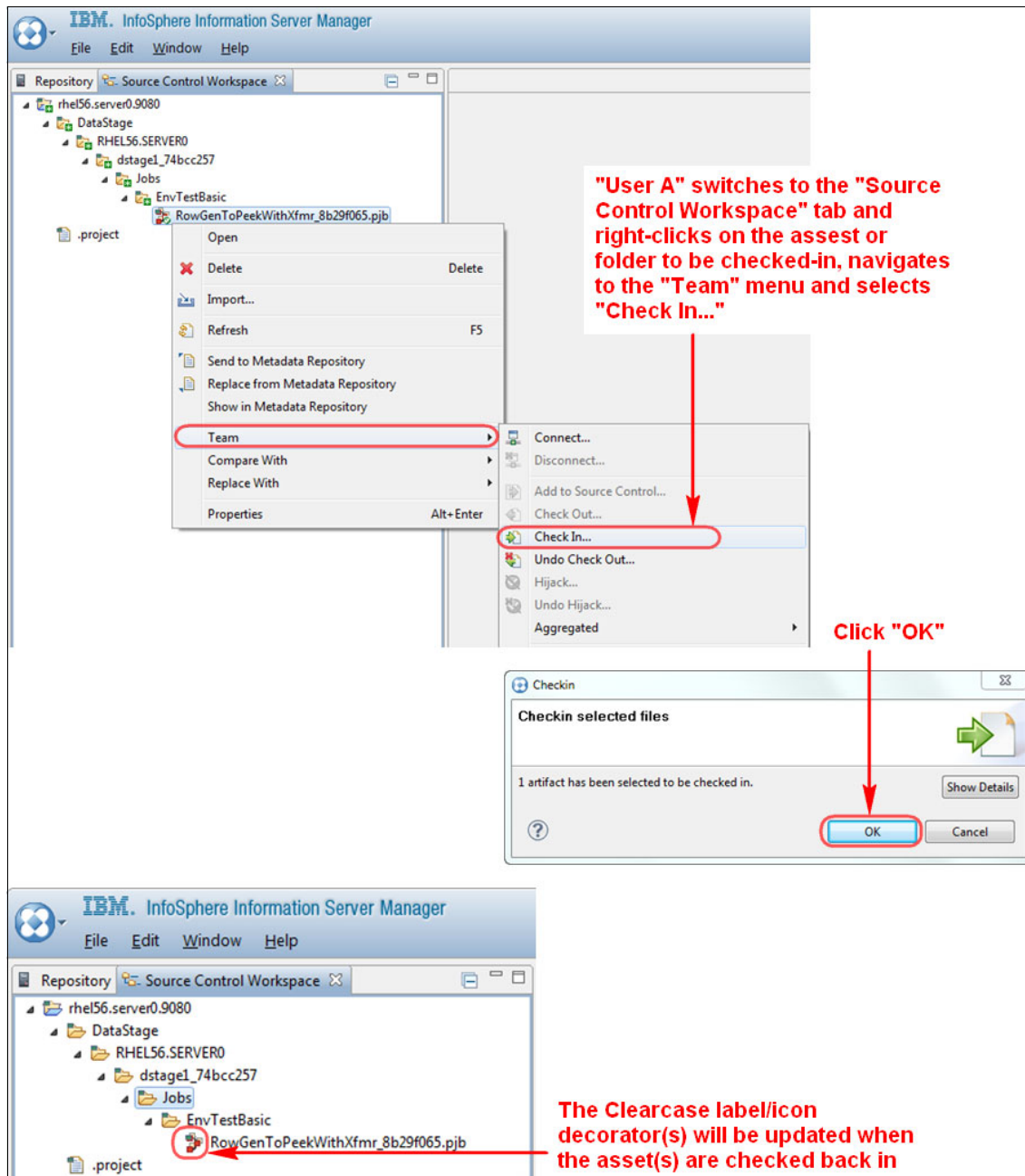


Figure 4-56 Checking in an asset with InfoSphere Information Server Manager Source Control Workspace view through Team menu

The user can specify additional options while checking in an asset by clicking **Show Details** in the **Checkin selected files** window (the intermediate step in Figure 4-56 on page 144) during the check-in process, as shown in Figure 4-57.

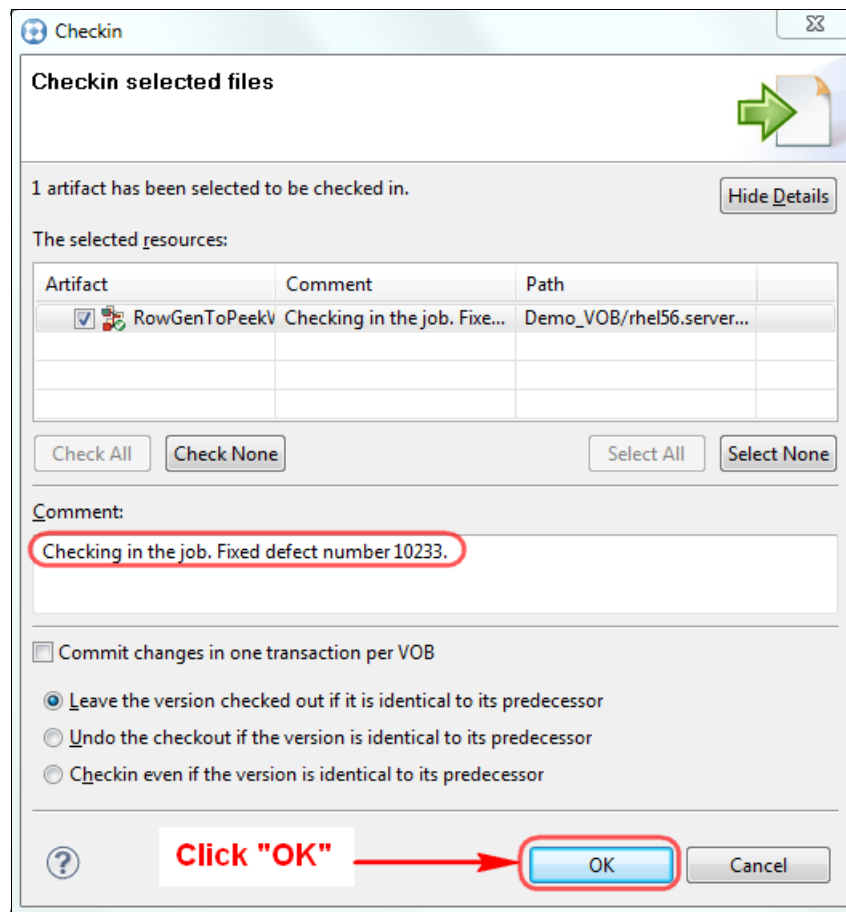


Figure 4-57 Specifying additional options during the check-in process

A standard practice is to provide comments during check-in and check-out processes of assets from a source control system.

4.19 Refreshing the Workspace view

As multiple users check assets in and out, the local workspace for a given user might miss assets that were checked in by other users.

This section describes the procedure to update the local source control workspace, with the most current state from the ClearCase repository. That procedure is depicted in Figure 4-58 on page 147 and Figure 4-59 on page 148.

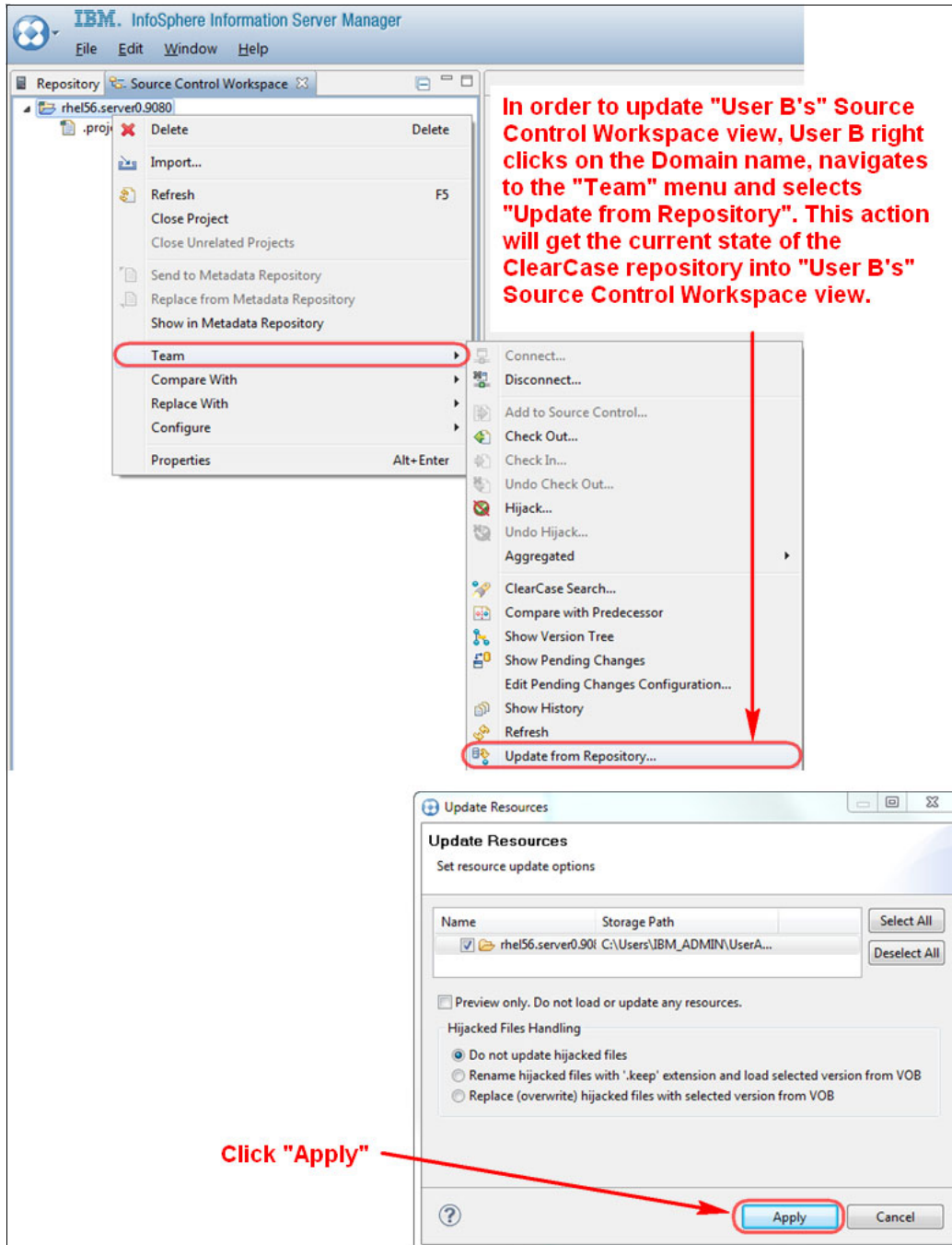


Figure 4-58 Refreshing the Workspace (step 1 of 2): Update from Repository

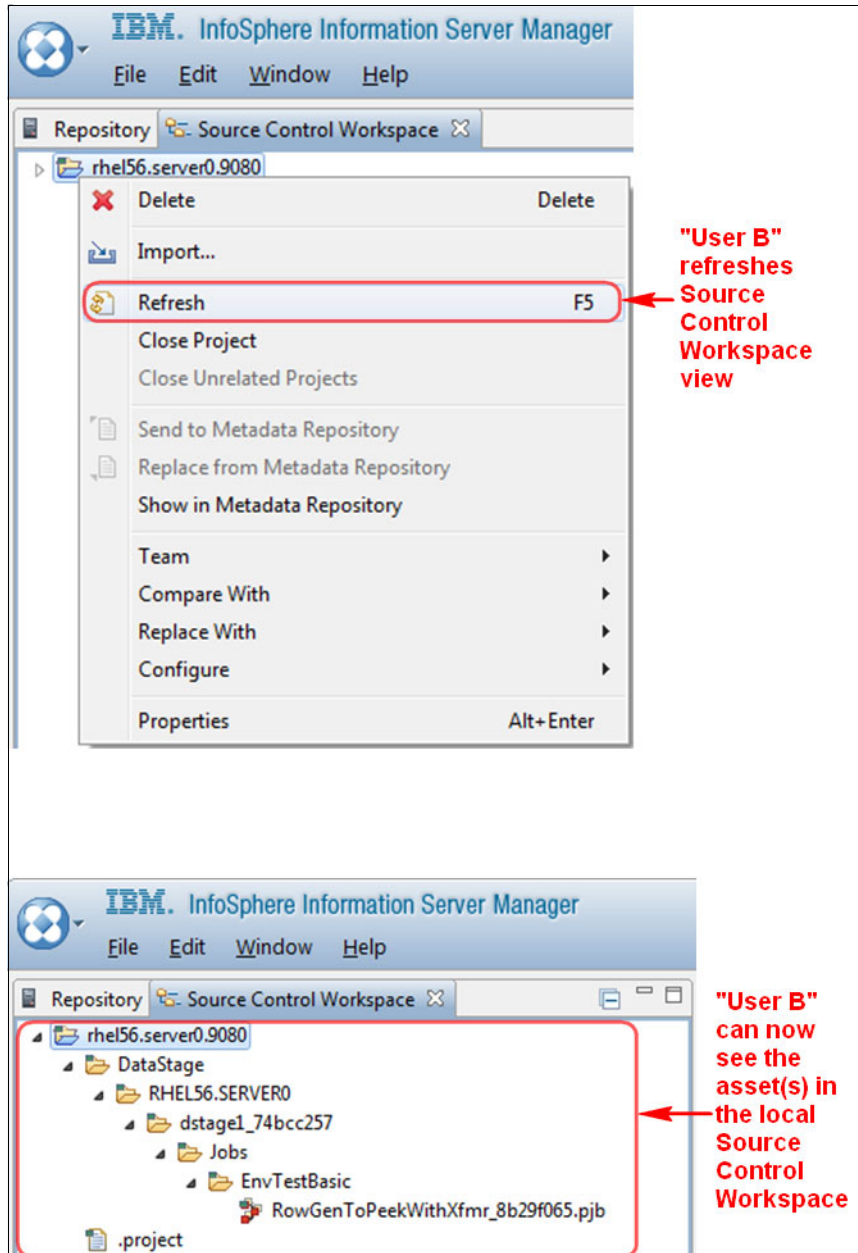


Figure 4-59 Refreshing the Workspace (step 2 of 2): Select the Refresh option

4.20 Creating and building deployment packages

This section describes the steps to create and build a package. They are depicted in Figure 4-60 on page 150 through Figure 4-62 on page 152. The creation of a package results in a definition. The build option actually builds the package, which then becomes available for deployment.

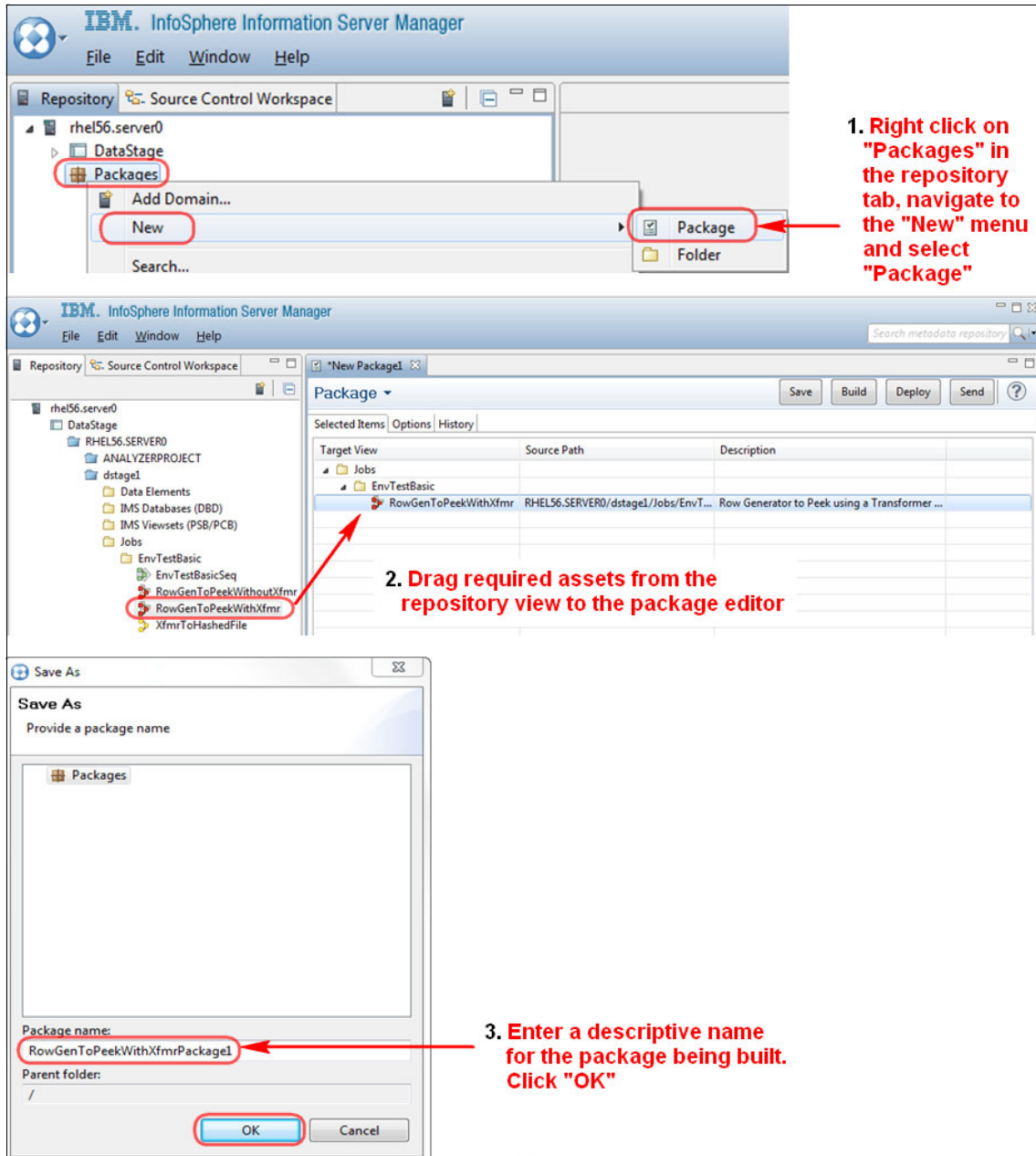


Figure 4-60 Defining and saving a package

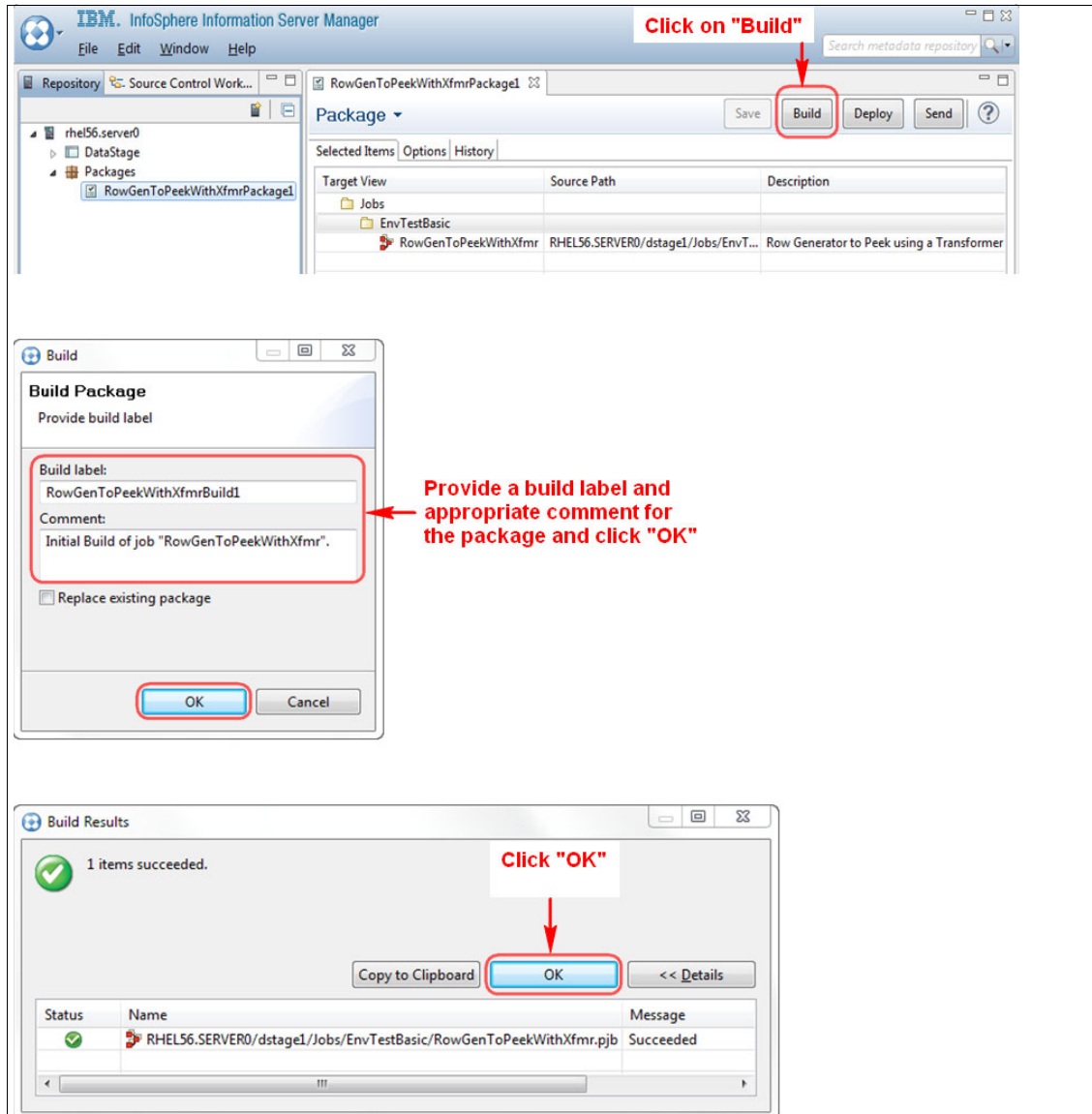


Figure 4-61 Building a package

Replica: When a package is built, the contents of the package (assets) are a replica of the version of the assets in the InfoSphere Information Server repository at the time of the build. They are not the version of the assets when the package definition was created. Be careful not to assume otherwise.

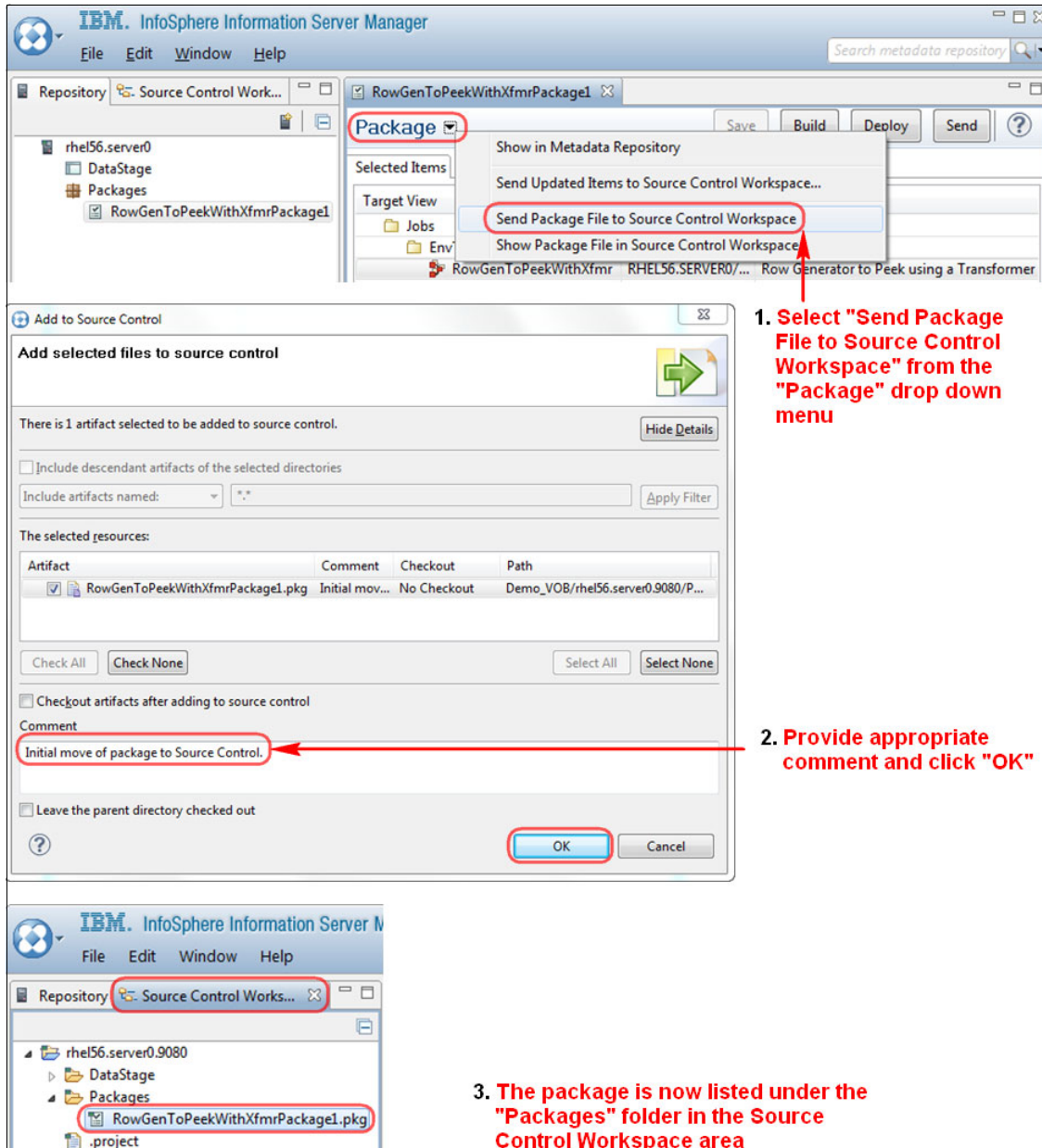


Figure 4-62 Sending the package to the source control workspace

4.21 Using the Deploy and Send options

The Deploy button, shown in Figure 4-63, works only for projects within the same InfoSphere Information Server domain.

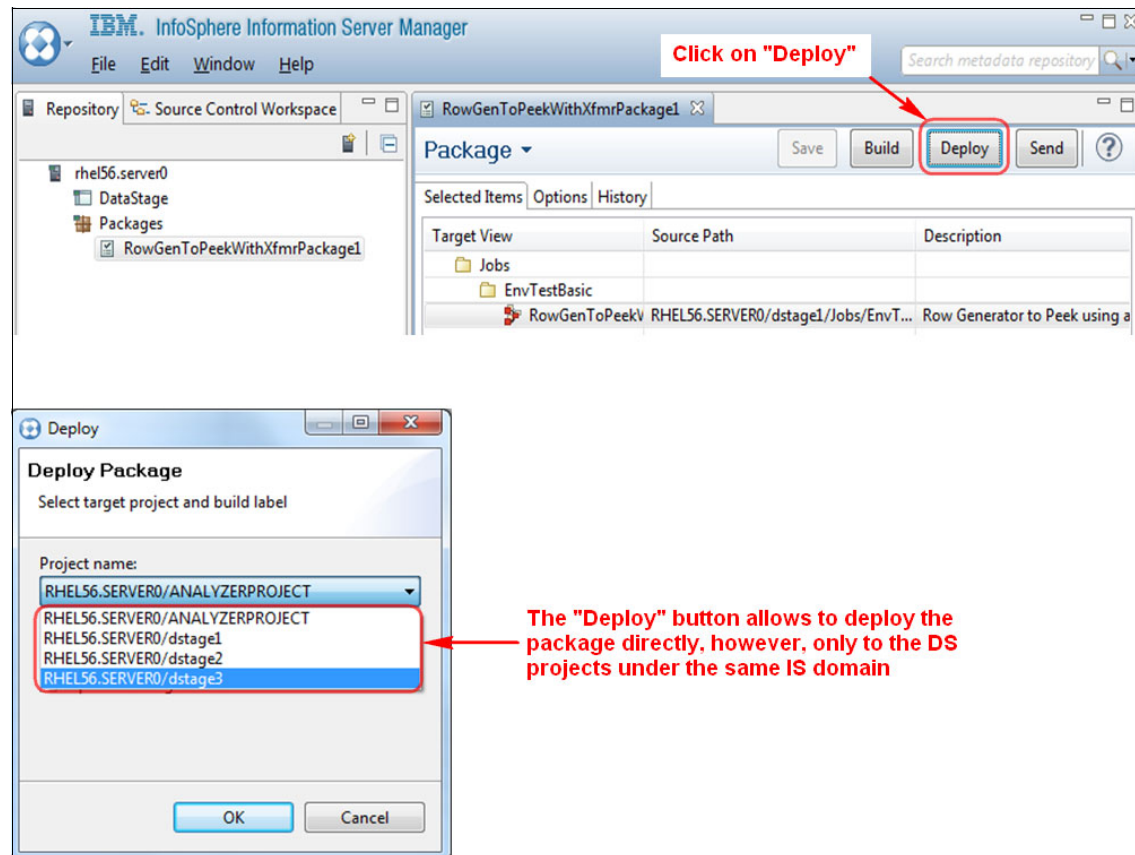


Figure 4-63 Using the Deploy button

To deploy a package to another domain, you may click **Send**, which saves the package to the local Windows file system. This file may then be placed in a public folder (either by means of a copy or FTP operation), which is accessible by release managers of other environments in the promotion path (test, QA, production).

The subsequent release managers use the **Deploy from file** option, which is described in 4.22, “Promoting deployment packages with ClearCase Explorer” on page 155 (Figure 4-67 on page 159 and Figure 4-68 on page 160).

Alternatively, you may rely on ClearCase for the promotion process (as described in 4.22, “Promoting deployment packages with ClearCase Explorer” on page 155). This way, deployment packages are not copied by FTP or exchanged in any way other than through ClearCase check-in and check-out capabilities.

Using the Send button is shown in Figure 4-64.

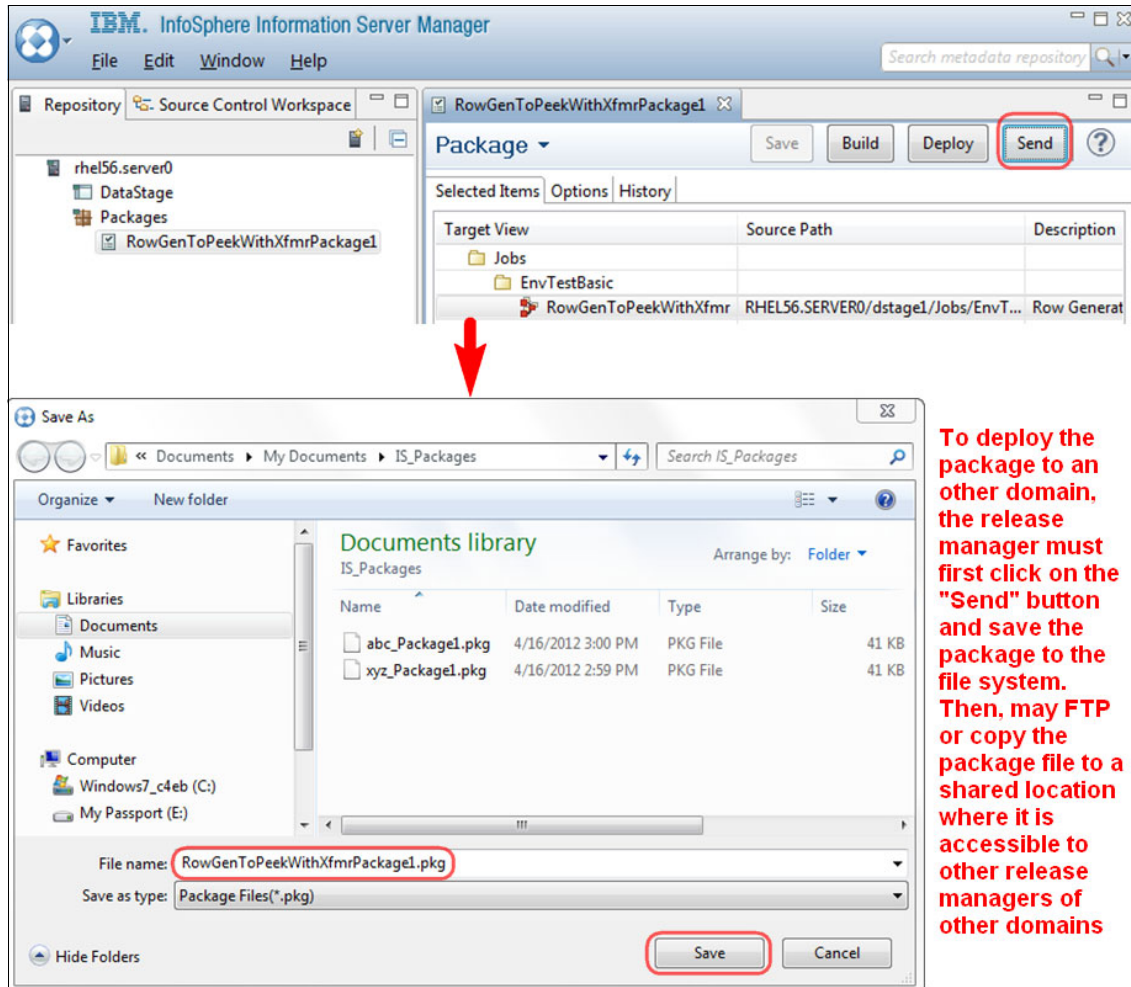


Figure 4-64 Using the Send button

Instead of using the Send button, use the approach in 4.21, “Using the Deploy and Send options” on page 153. This way, the entire promotion process (involving multiple release managers) follows the source control approach, relying entirely on the ClearCase check-in and check-out process.

4.22 Promoting deployment packages with ClearCase Explorer

The approach outlined in this section relies entirely on ClearCase for the package promotion process:

1. The development release manager still creates the package and sends it to the source control workspace as described previously (Figure 4-62 on page 152). This way, the package will be uploaded to ClearCase.
2. Test and production release managers must be granted access to the packages folder in ClearCase for the development environment. This way, those managers can check out deployment packages from ClearCase.
3. Test and production release managers, instead of checking out the packages using the InfoSphere Information Server Manager GUI (Repository view), check out the deployment packages by using the ClearCase Navigator in the ClearCase perspective. This step is described in 4.22.1, “Checking out deployment packages with ClearCase Explorer” on page 156. Those managers must be able to create views for the DEV folder structure in ClearCase.
4. When checking out, the deployment package is saved to the local workspace area.
5. Those release managers can use the **Deploy from file** option (in 4.22.2, “Deploying a checked out package” on page 158), navigating the local file system to locate the deployment package, and deploy it on their respective systems (test or production).
6. Because deployment packages are uploaded to ClearCase, repository ClearCase labels can be applied directly to the assets in the ClearCase repository. These labels can be used as an indication of the status of each deployment package, along the promotion process all the way from development, through test, and into production.

4.22.1 Checking out deployment packages with ClearCase Explorer

In this section, we describe the process of obtaining a package from a development environment, by using the ClearCase Explorer perspective in InfoSphere Information Server Manager. See Figure 4-65 and Figure 4-66 on page 157.

The assumption is that the test and production release managers have the permissions in ClearCase to create views on the deployment package assets relative to the development environment. In other words, those individuals may check in and check out deployment packages directly to and from ClearCase (without using the Repository or Workspace views in InfoSphere Information Server Manager, because they do not have credentials in the development InfoSphere Information Server environment to begin with).

They should not have ClearCase access to any other folders, (such as relative to jobs, table definitions, and so forth) for the development environment.

The permissions we refer to here are related to ClearCase. Again, TEST and PROD managers simply do not have access to the development InfoSphere Information Server instance at all. The only access they have is the ability to check in and check out deployment packages with ClearCase tools (such as the ClearCase Explorer perspective).



Figure 4-65 Opening the ClearCase Explorer perspective

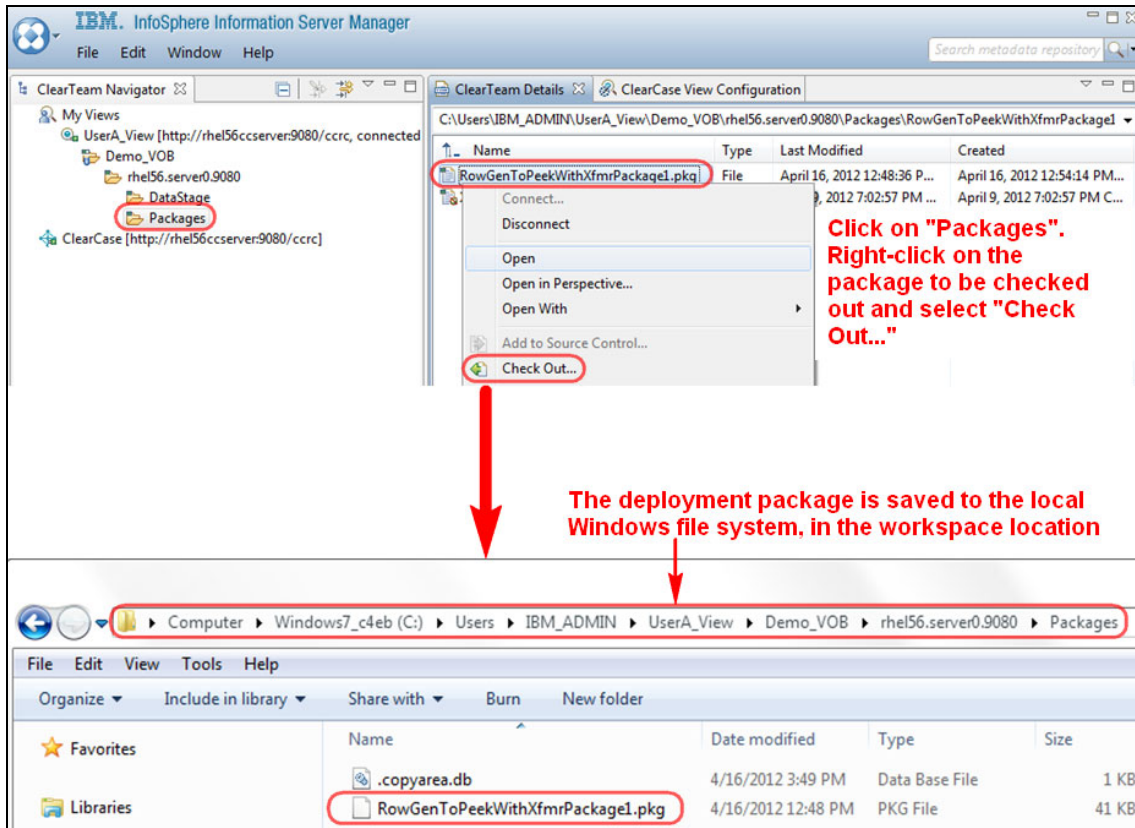


Figure 4-66 Checking out a package with ClearCase Explorer perspective

The view in Figure 4-66 is assumed to be for the DEV environment. The user following the steps in that illustration is a test or production release manager who is locating the package from DEV (again, retrieving it directly from the ClearCase repository). That person has permissions to check in and check out only packages (no other assets from DEV).

In ClearCase Navigator, the user navigates the view to locate the package folder, right-clicks the package name, and selects **Check Out**. This step will save the package to the local Windows file system where the test or product release manager is working.

The test or product release manager can then use the process described in 4.22.2, “Deploying a checked out package” on page 158, to deploy that locally saved package file.

4.22.2 Deploying a checked out package

This section describes the steps to deploy a package that was saved locally. This section is a continuation of the process that was started in 4.22.1, “Checking out deployment packages with ClearCase Explorer” on page 156, when the package was checked out of ClearCase and saved locally.

In Figure 4-67 on page 159, the test or production release manager is already connected to the target test or production environment. The process involves the following steps:

1. Right-clicking **Packages** in the Repository view, for the given target domain,
2. Selecting **Deploy Package From** → **File**.
3. Selecting the folder and file name
4. Clicking **Open**.

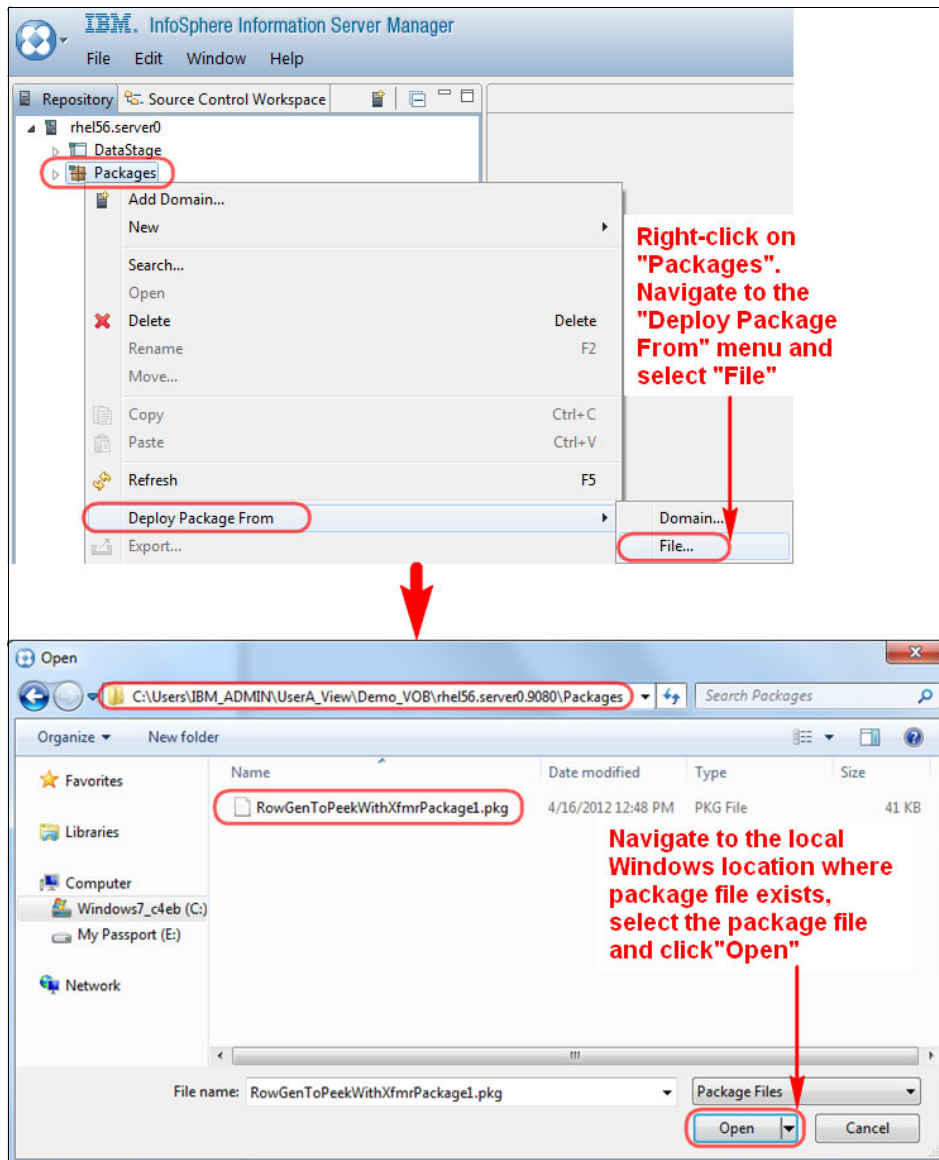


Figure 4-67 Deploying a package checked out through ClearCase Explorer: Selecting the package file

The continuation is illustrated in Figure 4-68 on page 160. After the package is uploaded to the Repository view of the target domain in InfoSphere Information Server Manager, click **Deploy**. In the pop-up window, select the target DataStage project, click **OK**, and the assets are imported into that DataStage project.

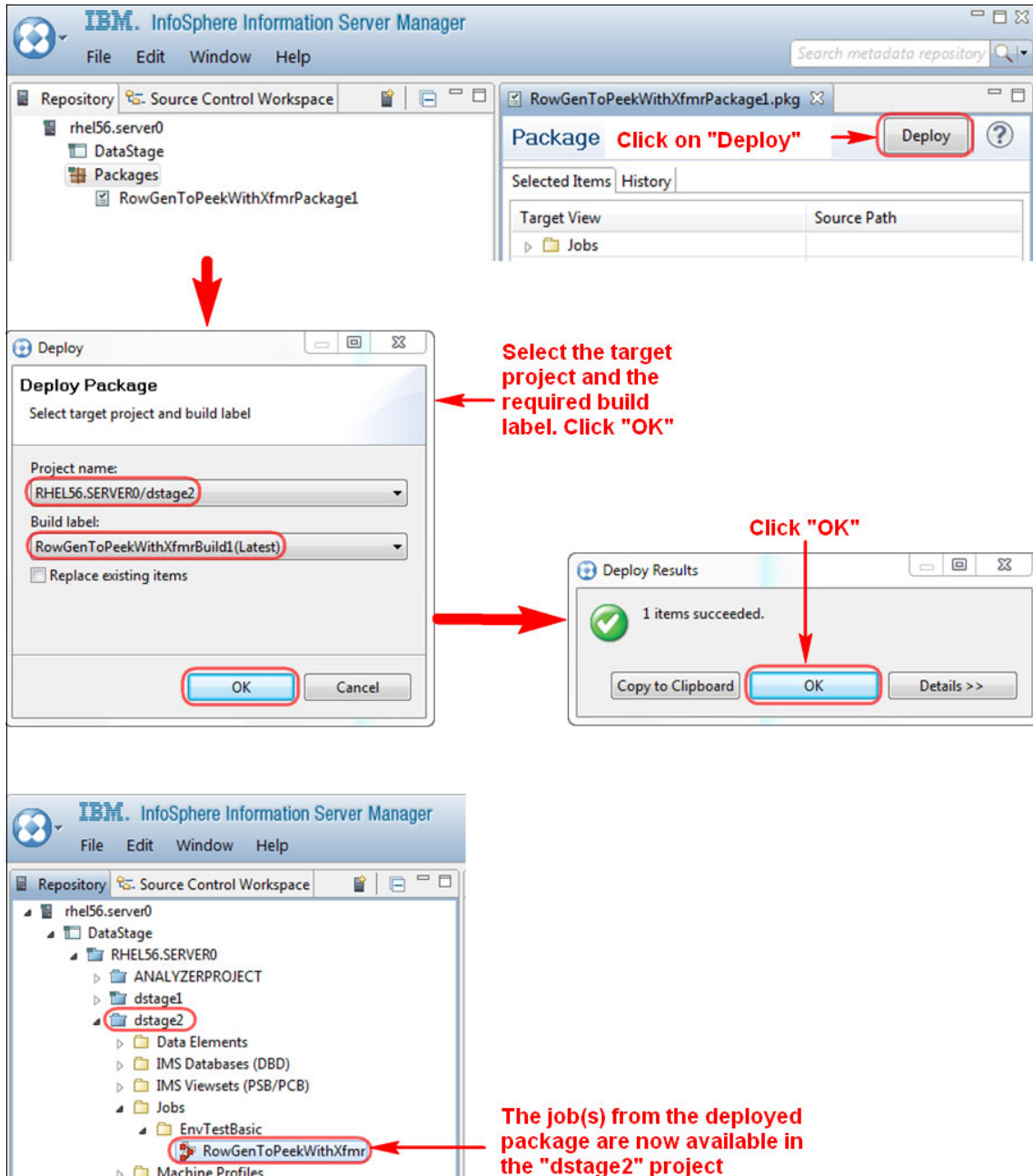


Figure 4-68 Deploying a package checked out through ClearCase Explorer: Doing the deployment

4.23 Test data deployment

Prior sections of this book describe the requirements for several InfoSphere Information Server environments, which are used through the information integration project lifecycle. For example, four distinct InfoSphere Information Server installations cover development, test, preproduction, and production. Within a single installation, several DataStage, QualityStage, and Information Analyzer projects can exist.

For each environment, organizations need to consider the provisioning of physical source and target databases that contain the relevant test data that is to be used by DataStage project (or projects) in that environment, as applicable. Typically, at the phase of the integration project where the production source and target databases are known, the approach often is to provision point-in-time *copies* of the relevant source and target databases because they are known to be sufficient for the testing requirements of the lesser environments. The copies may be refreshed occasionally if test scenarios require more recent data.

In provisioning this type of test data, the following considerations arise:

- ▶ Sizing and relevance

Source and target databases should generally be hosted on separate infrastructure from the InfoSphere Information Server and often separated by environment

Therefore consider a trade-off of the physical host system sizing requirements of taking full copies of production databases or if a subset will be able to deliver the test needs. The advantage of a subset is a lower physical sizing impact; the disadvantage is that the method of selection of the subset would have to be designed.

Two examples of methods of selection of data, with the objectives of taking a subset might be taking data that has timestamps younger than a given age, or copying a relevant subset of all the tables in the production database.

With any method that reduces data, an essential consideration is to be sure that the subset does not exclude relevant scenarios. However, this approach in turn can be of concern during initial parts of the information integration project lifecycle, when data is still being understood. This approach can exclude data that was not understood to be relevant until late in the development cycle, and as a result affect the project cost.

Where source data and rules are not well understood (which is the common case), profiling the data using tools, such as Discovery, is important.

In a *preproduction* environment, which might strive to reproduce production performance in controlled performance testing, full volume copies of the source and targets database might be unavoidable.

► Sensitive data issues

Production customer environments typically hold sensitive data of various types. For example data that is commercially sensitive has customer data protection concerns, credit card numbers, or data with other special or security considerations, such as being compliant with government standards for data privacy.

For non-production environments in which provisioning copies of production databases is required, data masking techniques can be used to assist with the management of the associated issues. *Data masking* is the technique of obscuring specific data elements in data stores.

Consider the access and location of the physical masking solution, which needs access to the sensitive data that is required to be masked.

Note that a characteristic of each technique is that the process should be repeatable because it might be required to be reapplied during test-data refresh, for example.

In selecting or designing a masking solution, the following issues arise:

- Data that is part of a relational key must be changed consistently with the needs of parent-child relationships.
- Specific patterns, field lengths, and substrings and type frequencies might contain embedded data that is relevant to the design of the subsequent DataStage jobs, quality rules to be developed, or data-profiling activities.

The following techniques are available to approach data masking within InfoSphere environments:

– Custom DataStage jobs

DataStage jobs can be coded to the rules that are required, by using the standard features available in DataStage as a custom development. Considering the previously described issues or complexity of particular requirements, this way might be a significant endeavor in itself notwithstanding the efficiency of DataStage development. (In generating arbitrary fresh or bulk test data, the row generator DataStage is also used.)

– IBM InfoSphere DataStage Pack for Data Masking

This pack is an add-on pack for DataStage that enables masking of the enterprise data by applying a data-masking policy. This pack provides an additional stage on the DataStage designer canvas and an associated masking policy editor.

In addition to the definition of custom policies, the following masking policies are available:

- Credit card number data masking policy

This policy generates an appropriate mask for credit card numbers, based on the source data. IBM InfoSphere DataStage Pack for Data Masking supports data masking for American Express, MasterCard, Visa, and Discover credit cards, as examples.

- Email address data masking policy

This policy generates an appropriate mask for source email addresses. You can mask the entire email address, only the user name, or only the domain name.

- US national ID data masking policy: National ID (US)

The national identification number for USA is the Social Security number. This policy generates an appropriate mask for the Social Security numbers, based on the source data.

- Canada national ID data masking policy: National ID (CA)

The national identification number for Canada is the Social Insurance Number. This policy generates a valid Canada Social Insurance Number, based on the source data.

- French national ID data masking policy: National ID (FR)

The national identification number for France is the French National Institute for Statistics and Economic Studies number. This policy generates a valid French National Institute for Statistics and Economic Studies number based on the source data.

- Italy national ID data masking policy: National ID (IT)

The national identification number for Italy is the Fiscal Code. This policy generates a valid Fiscal Code number based on the source data. When the Fiscal Code number is masked, the part containing the name is copied from the source and the other parts are masked. The result is always the same for separate runs of the same source data.

- Spain national ID data masking policy: National ID (ES)

The national identification number for Spain is the Fiscal Identification number (NIF) or the Foreigner's Identification number (NIE). The NIF is given to citizens of Spain; the NIE is given to foreign residents.

- UK national ID data masking policy: National ID (UK)

The national identification number for the UK is the National Insurance Number (NINO). This policy generates a valid National Insurance Number based on the source data.

- Date age data masking policy
This policy generates a new date based on the source data value. The date age data masking policy does not generate random dates.
- Repeatable replacement data masking policy
This policy masks source data in the format of the source data. You must always provide input data to the repeatable replacement data masking policy. You can use this data masking policy to convert keys such as the primary key or the foreign key.
- Random replacement data masking policy
This policy masks source data in different formats for different runs of the source data.
- Hash lookup data masking policy
This policy masks input columns by using a reference table on a database. It calculates hash value based on the value of a source column, and retrieves a record whose hash key column matches the hash value.
- Other masking possibilities include but are not limited to masked decimal conversions, masked character and masked left and right conversions.

Figure 4-69 on page 165 shows a simple DataStage job where the Data Masking pack was installed as part of the InfoSphere Information Server installation that is using the Data Masking stage. This job reads its input and writes its output to sequential files, but it can just as easily be reading and writing from supported source and target databases by using the relevant stages. The Data Masking stage can optionally have a reject link as shown Figure 4-69 on page 165. In that case, where source records do not match the expected patterns, the row can be rejected for further investigation.

A data masking DataStage job supports connectivity to the wide variety, types, and versions of source and target databases that are supported by DataStage.

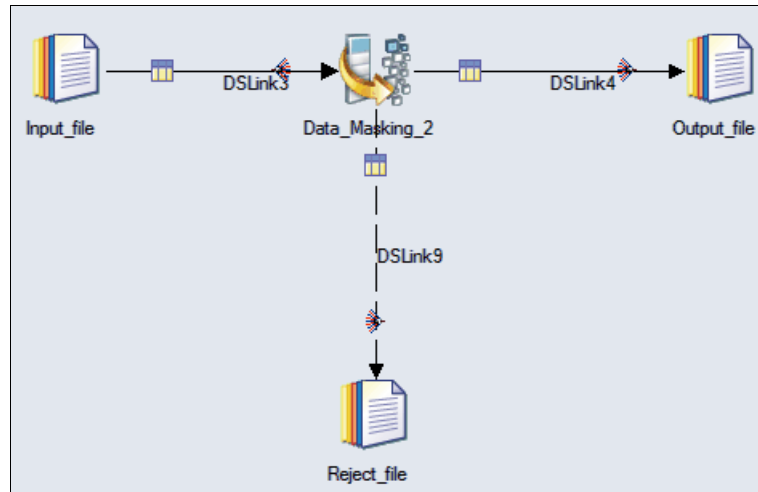


Figure 4-69 Data masking stage in DataStage

Within the data masking stage, the data masking policy editor can be used to separately set the masking rules for each column as required.

– Use IBM InfoSphere Optim™ Data Masking Solution

Optim is an additional suite, available under the InfoSphere brand, that is specifically aimed at data masking. It is briefly described here, in the context of the enterprise solution to data masking.

With InfoSphere Optim Data Masking Solution, users can apply various proven data transformation techniques to replace sensitive real data with contextually accurate and realistic fictitious data. Users can mask data in a single database or across multiple related systems. Simple examples of the masking techniques in InfoSphere Optim include substrings, arithmetic expressions, random or sequential number generation, date-aging and concatenation. And its context-aware masking capabilities help ensure that masked data retains the look and feel of the original information. Those capabilities make it easy to de-identify many types of sensitive information, such as birth dates, bank account numbers, street address and postal code combinations, and national identifiers.

InfoSphere Optim provides a scalable data masking solution with flexible capabilities that can be easily adapted to your current and future requirements. InfoSphere Optim supports all leading enterprise databases and operating systems, including IBM DB2, Oracle, Sybase, Microsoft SQL Server, IBM Informix®, IBM IMS, IBM Virtual Storage Access Method (VSAM), Teradata, Adabas, Microsoft Windows, UNIX, Linux, and IBM z/OS®. In addition to providing data management support for all custom

and packaged applications, InfoSphere Optim has the meta-model knowledge to support most of the current enterprise resource planning (ERP) and customer relationship management (CRM) applications in use today. Examples of those are SAP, Oracle E Business Suite, PeopleSoft Enterprise, JD Edwards EnterpriseOne, Siebel, and Amdocs CRM. It also supports compliance with privacy regulations, such as HIPAA, GLBA, DDP, PIPEDA, Safe Harbour, PCI DSS, and others.

4.24 Lifecycle management of other assets

Previous sections describe lifecycle management of DataStage and QualityStage assets.

InfoSphere Information Server assists in the management of many assets that arise in the Information Integration landscape of which DataStage and QualityStage assets are just a part.

Table 4-2 lists the extensive variety of assets, grouped by category, that InfoSphere Information Server can manage.

Table 4-2 Types of InfoSphere Information Server assets

Asset category	Asset
Business Glossary	<ul style="list-style-type: none"> ▶ Categories ▶ Terms
Fastrack	<ul style="list-style-type: none"> ▶ Mapping components ▶ Mapping compositions ▶ Mapping specifications ▶ Project templates ▶ Projects ▶ Role assignments, reports, common metadata, glossary assets, and IBM InfoSphere DataStage and QualityStage assets that are associated with a project
Information Analyzer	<ul style="list-style-type: none"> ▶ Projects, including the assets that are associated with them, such as analysis results, data rules, and data classes ▶ All data classes, regardless of which project they are associated with ▶ Reports and common metadata that are associated with a project

Asset category	Asset
Common metadata assets	<ul style="list-style-type: none"> ▶ Business intelligence (BI) assets: <ul style="list-style-type: none"> BI models BI collections Cubes BI reports BI report queries ▶ Implemented data resources: <ul style="list-style-type: none"> Host computers Databases Database schemas Stored procedures Database tables Data files Data file structures Data connections Data item definitions ▶ Logical data model assets: <ul style="list-style-type: none"> Logical data models Logical entities Logical relationships Entity generalization hierarchies Logical domains Subject areas Diagrams ▶ Physical data model assets: <ul style="list-style-type: none"> Physical data models Design tables Design stored procedures Physical domains ▶ Miscellaneous assets <ul style="list-style-type: none"> Data connections Custom attributes Contact libraries
Reporting assets	<ul style="list-style-type: none"> ▶ Reports ▶ Report results
Security assets	<ul style="list-style-type: none"> ▶ Users, with or without roles, credentials, and credential mapping ▶ Groups, with or without roles

Asset category	Asset
DataStage and QualityStage assets (described in prior sections)	Custom folders (external files in the project directory) <ul style="list-style-type: none"> ▶ Data connections ▶ Data elements ▶ Data quality specifications (lookup tables, rule sets, and match specifications) ▶ IMS databases ▶ IMS viewsets ▶ Jobs (mainframe, parallel, sequence, and server) ▶ Job executables ▶ Machine profiles ▶ Parameter sets ▶ Routines (mainframe, parallel, and server) ▶ Shared containers (parallel and server) ▶ Stage types ▶ Table definitions ▶ Transforms ▶ Shared tables and related common metadata assets that are associated with table definitions

In reviewing Table 4-2 on page 166 and comparing the lifecycle of metadata-related assets, in particular, against the lifecycle of DataStage and QualityStage assets, an important concept to recognize is that metadata has a much longer lifecycle than DataStage and QualityStage.

Consider the following examples:

- ▶ Some Business Glossary terms will be applicable for as long as the duration of the owning organization.
- ▶ Source and target database table technical metadata must be defined and exist prior to any DataStage project that will extract from it. In fact, throughout the lifecycle of a DataStage project, many versions of a DataStage job might be concerned with reading from that same table.
- ▶ Information Analyzer data quality rules can exist in isolation of any need to develop a regular ETL solution in DataStage.

Chapter 5, “Metadata deployment architectures” on page 169 describes the InfoSphere Information Server deployment topology choices that take into account the wider information integration assets as described in Table 4-2 on page 166.



Metadata deployment architectures

All InfoSphere Information Server product modules generate and consume metadata. The generated metadata is persisted in the InfoSphere Information Server repository. Through this shared InfoSphere Information Server repository, InfoSphere Information Server provides collaboration between the product modules and users, such as analysts, data stewards, subject matter experts, and developers. This behavior is true for all InfoSphere Information Server instances, such as development, test, production, and other environments that a company might have. However, the development lifecycle varies for different InfoSphere Information Server assets, such as DataStage job design (code), as opposed to data source technical metadata.

The various lifecycles of the InfoSphere Information Server assets can affect the deployment architecture for the InfoSphere Information Server product modules. Depending on business requirements, InfoSphere Information Server product modules might be deployed in some or all of the environments. The same product module can also be deployed differently, based on the feature you are using and the level of metadata collaboration you are leveraging. Therefore, the typical development (DEV), test (TEST), and production (PROD) deployment architecture might not be appropriate for each of the InfoSphere Information Server product modules and business use case scenarios.

You must consider various factors when you choose the deployment architecture for InfoSphere Information Server. To effectively use the InfoSphere Information Server product modules and achieve the business objectives, this chapter introduces the following four deployment architectures.

- ▶ SIMPLE
- ▶ UNIFIED
- ▶ REPORTING
- ▶ GOVERNANCE

In the following sections we describe which architecture might fit your case, the implementation steps, and the advantages and disadvantages of each architecture.

5.1 SIMPLE architecture

SIMPLE is the typical DEV, TEST, PROD environment architecture. In a typical deployment architecture, the source code is developed in the DEV environment, tested and deployed to the TEST environment, and then tested and deployed to the PROD environment. This process is commonly done by using a type of source code control system. The deployment to the PROD environment is done in accordance with the production administration and scheduling, so that the main process on PROD will not be affected, in any way, by the deployment.

Development of DataStage and QualityStage fits into this typical development lifecycle. An important consideration is that the PROD environment must not be negatively affected by any other processes, because the main purpose of the PROD environment in this case is *data integration*.

When you use other InfoSphere Information Server product modules, such as IBM Information Analyzer (IA), FastTrack, Business Glossary, and Metadata Workbench with DataStage and QualityStage, When you use other InfoSphere Information Server product modules, such as IBM Information Analyzer (IA), FastTrack, Business Glossary, and Metadata Workbench with DataStage and QualityStage, you must consider whether you need to dedicate separate capacity from the PROD environment data integration process for these other modules.

Deciding which architecture to choose

If you are using only DataStage or QualityStage, the common practice is to choose this SIMPLE architecture. Information Services Director will be handled the same as DataStage jobs.

Even if you are using other InfoSphere Information Server product modules, such as Information Analyzer, FastTrack, Business Glossary, and Metadata Workbench, there are also times when this SIMPLE architecture is the appropriate architecture. In this case, consider the following questions to help you identify whether this SIMPLE architecture is appropriate for your case:

- ▶ Are you reusing the metadata generated in any of the InfoSphere Information Server product modules for any other purpose? Is there collaboration?

Examples are as follows:

- Information Analyzer profiling results published and reused in DataStage job development
- Linking technical assets to business terms
- ▶ Is your execution of data integration jobs in the PROD environment affected by any of the following items?
 - Reporting metadata with Metadata Workbench
 - Sharing glossary using Business Glossary
 - Profiling data or using quality rules as factors in Information Analyzer (Rules Stage used in DataStage jobs for data integration are not included, because it is a part of a DataStage job)

The main purpose of the PROD environment is data integration, so it should not be affected by any of the other processes.

If any of these points match your case, then the SIMPLE architecture is probably *not* the correct architecture choice for you. Consider using either the UNIFIED or REPORTING architecture.

Typical configuration

As shown in Figure 5-1, the typical configuration for this SIMPLE architecture is a DEV, TEST, PROD three-tier configuration.

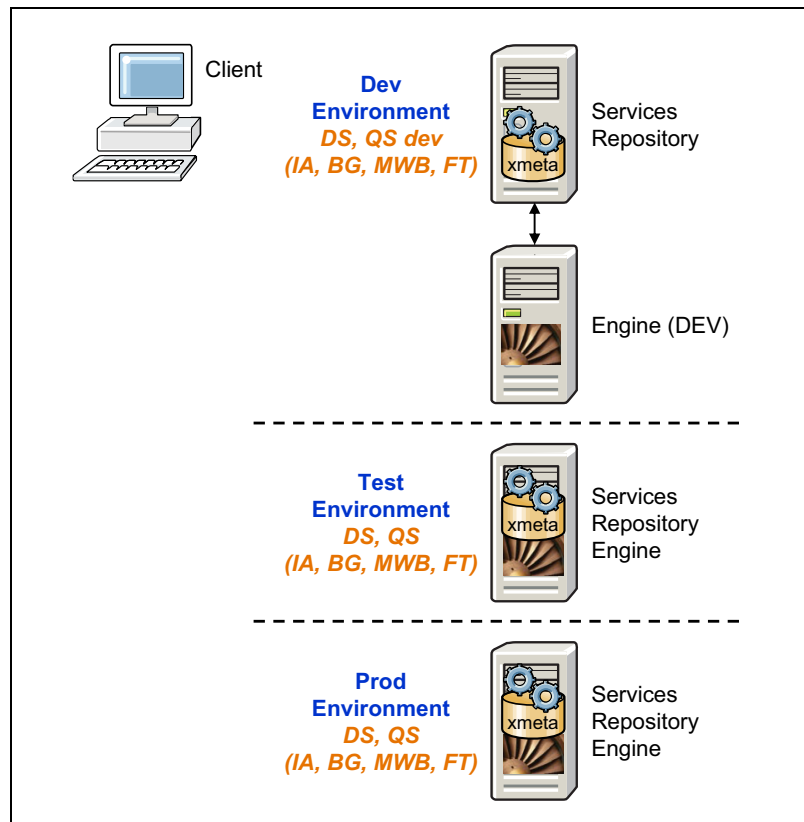


Figure 5-1 SIMPLE architecture three-tier configuration

Topology

For the DEV environment, a three-tier configuration is typical. The separation of the resources of the services and the engine tiers enables optimization and scalability for the specific resources required by the developers on the engine tier, while allowing the services tier to manage its load independently.

For the TEST and PROD environments, most customers use a standard two-tier configuration. Details of two-tier configuration is in 2.4 “Client server installation topology” on page 24 and 2.7 “InfoSphere Information Server clustered installations” on page 28.

Implementation

Figure 5-2 shows an example of a deployment process for the three-tier configuration.

DataStage and QualityStage jobs follow the typical DEV, TEST, and PROD development lifecycle model. The following steps are the basic process:

1. Jobs are developed, then tested in the DEV environment.
2. Jobs are deployed to the TEST environment, then tested.
3. Jobs are deployed to the PROD environment.

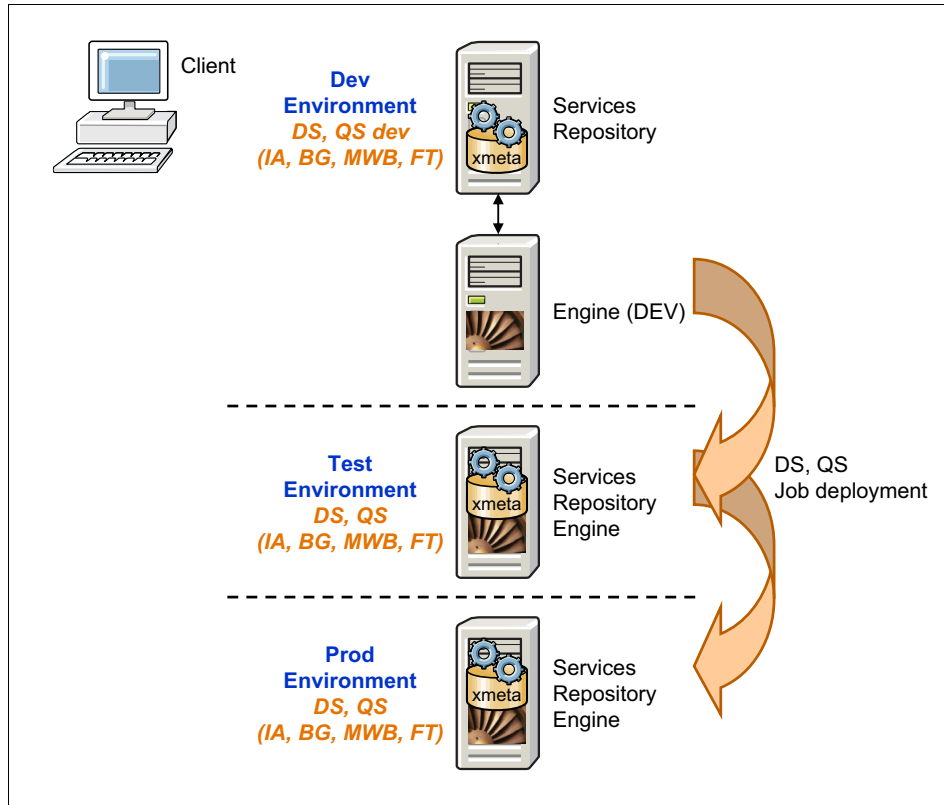


Figure 5-2 Three-tier configuration deployment process

Advantages and disadvantages

There are advantages and disadvantages for this SIMPLE architecture.

Advantages

All generated metadata (technical, operational, business) will already exist in the PROD environment, so you do not need to copy any metadata to a different environment.

Disadvantages

If the metadata reporting or data profiling becomes more than minimal, it can impact the PROD data integration process.

UNIFIED architecture: When data integration in the PROD environment is affected by other processes, consider using the UNIFIED architecture.

5.2 UNIFIED architecture

The InfoSphere Information Server shared InfoSphere Information Server repository is used for collaboration and all metadata reporting, including operational and business metadata. The UNIFIED architecture is a combination of development and the metadata reporting and collaboration environment.

A significant driver for establishing this UNIFIED architecture is that shared business metadata (such as for profiling and business rules) does not necessarily follow the standard software development lifecycle as described in 4.24 “Lifecycle management of other assets” on page 166. Some operations (such as profiling) can be used in support of development activities. Furthermore, information security requirements might call for a separation of processing engines but still have the need to share metadata (as explained in 5.4 “GOVERNANCE architecture” on page 191).

Although the collaboration and reporting are minimal, the SIMPLE architecture is the suggested choice, with the metadata product modules deployed in the PROD environment. The question still remains, what is the best environment to provision metadata reporting and collaboration functionality? Given the “production” nature of the metadata-based reports, you might consider the PROD environment (as in the SIMPLE deployment scenario). However, the main purpose of the PROD environment is to integrate the “data” that is to be used by downstream applications. Heavy use of Business Glossary, regular processing of Data Quality rules, generating lineage and impact reports does not fit the “data” paradigm of the PROD environment. Furthermore, there are no developers with whom to “collaborate” in the PROD environment. Therefore, for this use case,

selecting a different environment in which to deploy these other product modules is necessary.

The most beneficial results can be achieved by repurposing the DEV environment to process Data Quality rules. Use production data on a processing engine in the DEV environment that is completely separate from the development engine, thereby ensuring data privacy and security through this separation. In this manner, the *former* DEV environment serves multiple purposes, yet is still fit for each individual purpose with all their separate functional and non-functional requirements. In this situation, the environment is no longer a SIMPLE architecture, but rather a UNIFIED architecture because it serves multiple purposes. The UNIFIED architecture is one that supports collaboration, and expanded data and metadata reporting.

By creating this UNIFIED architecture, you will be able to dedicate the PROD environment to its main purpose of *data integration*, and it will not be affected by any other functionality.

Deciding which architecture to choose

If you are using Information Analyzer, FastTrack, Business Glossary, and Metadata Workbench with DataStage, QualityStage, and Information Services Director, the common practice is to consider this UNIFIED architecture.

But there are also other cases, when another deployment architecture is suitable even if you are using those product modules. For example, if you do not need dedicated resources for these product modules or collaboration with developers, the SIMPLE architecture might be more appropriate. If you have massive reporting requirements, the REPORTING architecture (described in 5.3 “REPORTING architecture” on page 185) might be more appropriate.

Typical configuration

As Figure 5-3 shows, the typical structure for this architecture is a UNIFIED DEV, TEST, and PROD three-tier environment.

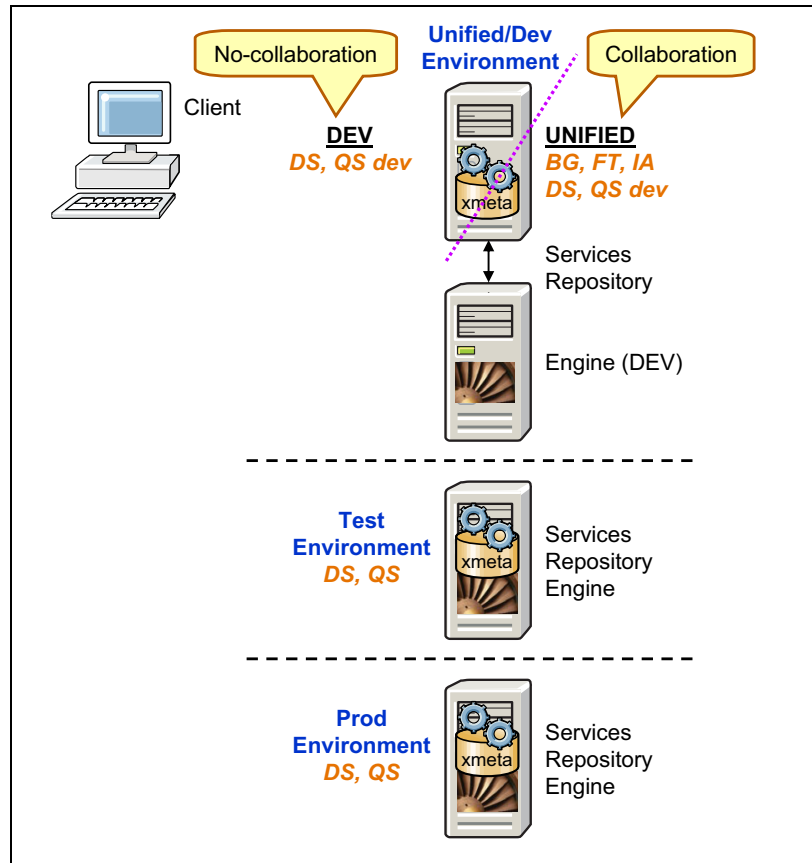


Figure 5-3 UNIFIED architecture three-tier environment

In the UNIFIED architecture, the DEV environment is not only used for the development of DataStage and QualityStage jobs, but is also used as an environment for metadata reporting and collaboration. The point that the DEV environment now serves multiple purposes is the main difference between the SIMPLE architecture and the UNIFIED architecture. This reason is why the DEV environment becomes the UNIFIED DEV environment.

TEST and PROD environments are useful only for execution environments, which are for DataStage and QualityStage jobs.

Topology

For the UNIFIED/DEV environment, a three-tier configuration is typical, as in a DEV environment. The reason is because the compiling and testing by many concurrent users requires capacity. You can also have multiple engines under the InfoSphere Information Server repository, which enables you to isolate the DataStage and QualityStage development away from the common shared metadata processes.

When you use Information Analyzer, having an engine that is dedicated to Information Analyzer is meaningful, as shown in Figure 5-4.

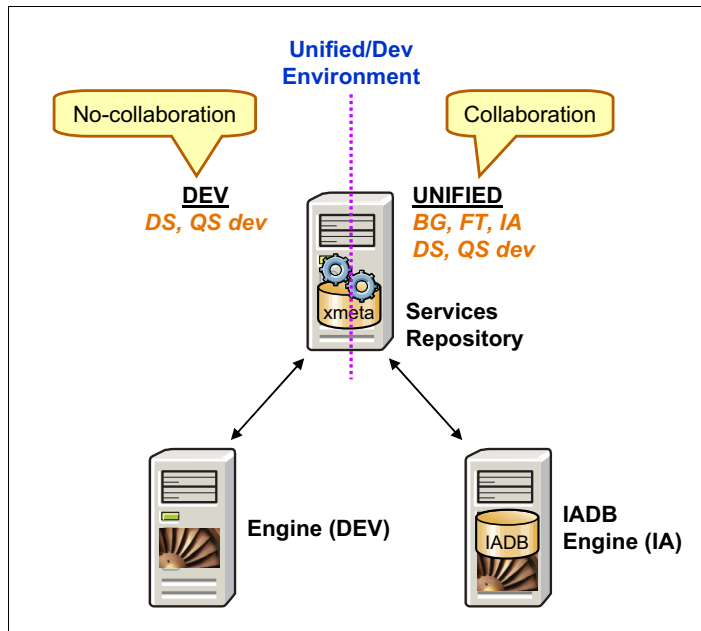


Figure 5-4 UNIFIED architecture with Information Analyzer

Having a dedicated engine enables you to connect to and process production data securely. The usage of each engine node is as follows:

- ▶ Development engine node: With this engine node, developers can develop, compile, and test jobs on that given server.
- ▶ Information Analyzer engine node: This engine node is used for profiling data and data quality rules by connecting the production data. It has a stand-alone database (IADB) that enables you to secure data and also account for any performance considerations with the sampled data. Details for securing data is described in “Data profiling of confidential data” on page 182.

For TEST and PROD environments, most clients use a standard two-tier configuration because of security, usage, and availability reasons.

Implementation

Figure 5-5 on page 180 shows the implementation image. The basic implementation steps are as follows:

- ▶ DataStage and QualityStage jobs follow the typical DEV-TEST-PROD development lifecycle model.
- ▶ For metadata reporting purposes, you must copy DataStage and QualityStage job design and operational metadata, and external metadata into the UNIFIED architecture environment.
 - DataStage and QualityStage job design metadata is generated in the UNIFIED DEV environment. However, you must still copy the design metadata, matching the current PROD environment version from the DEV to the UNIFIED PROD environment, with the same naming convention.

Job design: FastTrack mappings and Information Analyzer Rules Stages that are used in DataStage jobs are included in this job design.

- DataStage and QualityStage operational metadata are generated to an XML file that is stored on the file system at job run time. This operational metadata can be imported into xmeta through a shell script (`RunImportStart.sh`) or a batch file (`RunImportStart.bat`) that is located on the engine tier of the PROD environment. Because the credential and connection for loading the metadata into the target xmeta is specified in a properties file (`runimport.cfg` or `runimport.cfg.unix`), you must specify the UNIFIED environment instead of the PROD environment for the target xmeta; the operational metadata will then be loaded into the UNIFIED environment. Details about managing the operational metadata is described in *IBM InfoSphere Information Server, Guide to Managing Operational Metadata*, which is at the following location:

<http://publibfp.boulder.ibm.com/epubs/pdf/c1934770.pdf>

Resolution: Operational metadata is effective when your design metadata is not enough for your metadata reporting. This situation can often happen when you use parameters without appropriate default values in your jobs. Operational metadata will resolve the values that are used in execution, so you will not need to stitch the metadata manually. For more information, see 3.2.3 “Operational metadata” on page 65.

- If you have a requirement to include external assets such as PL/SQL and COBOL programs in your metadata reporting, you will also need to generate that external metadata in the UNIFIED InfoSphere Information Server repository.
- ▶ Business metadata is generated and consumed in the UNIFIED environment, so promoting business metadata is not required.

Functionality: Business Glossary has its own work flow as part of its built-in functionality. Therefore, it does not require the same SDLC considerations as software development.

- ▶ Information Analyzer profiling results are generated and published in the UNIFIED environment, so no promoting is required.

Lifecycle: Information Analyzer has its own development lifecycle function rather than the typical one. The results of profiling will be published and be visible to other product module users, such as DataStage developers who need to understand concepts, such as potential data anomalies and data format, and how to code for them.

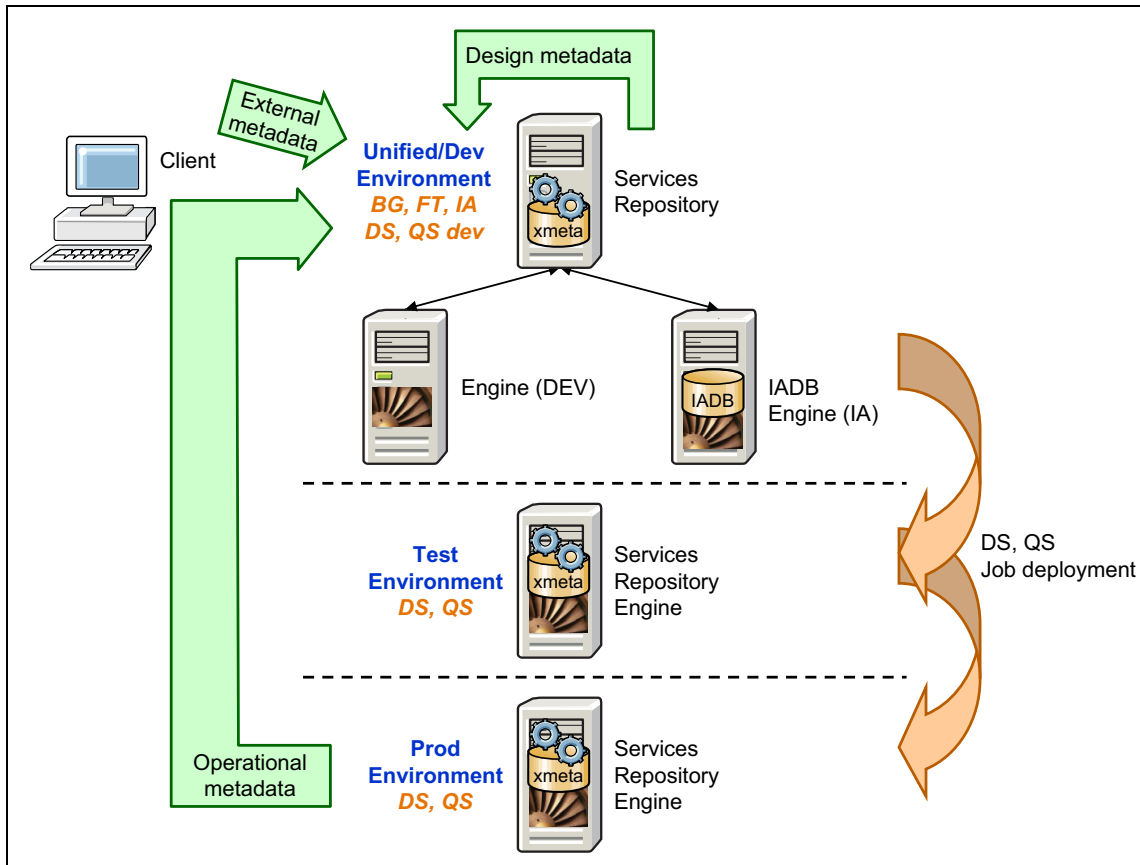


Figure 5-5 Implementation image of TEST and PROD environments

Advantages and disadvantages

There are advantages and disadvantages for this UNIFIED architecture.

Advantages

The following advantages apply:

- ▶ The PROD environment is minimally affected by the various metadata needs because metadata reporting is done in a separate environment.
- ▶ Metadata can be reused and collaboration is simplified because all product modules are located in the same environment. For example, business metadata is both generated and consumed by the Business Glossary and authors or users in the UNIFIED/DEV environment.
- ▶ The DEV environment is minimally affected by the Information Analyzer profiling because Information Analyzer has a separate engine.

- ▶ Production data is secure because the Information Analyzer engine and IADB is separated from the DEV environment.
- ▶ Because the UNIFIED/DEV environment is a three-tier configuration, additional server tiers can be added to better load-balance the metadata needs and the development. Details of the three-tier configuration is described in Chapter 2, “InfoSphere Information Server installation topologies” on page 19.

Disadvantages

The following disadvantages apply:

- ▶ A large number of users who use Metadata Workbench/Business Glossary or an intricate glossary can affect the performance of the DEV environment, and vice versa.

Resolution: Creating a new single REPORTING environment dedicated for reporting can solve this issue. See 5.3 “REPORTING architecture” on page 185.

- ▶ For operational metadata, the properties files might need slight modification to account for the fact that the metadata was generated in a separate environment.

Tip: A well-understood change management process must be in place for such an environment.

- ▶ Additional effort is required to copy the operational metadata to the UNIFIED environment.

Tip: Because the common UNIFIED InfoSphere Information Server repository shares all metadata, an environment where patches can be applied and tested, or test cases can be worked out without affecting the shared environment that all other users are using, is necessary. This environment should be a stand-alone “sandbox” environment.

Data profiling of confidential data

Data profiling is typically done against the production data. Information and data security requirements require strict controls over access to this production data, and isolating production execution environments can guarantee this behavior. In a UNIFIED environment, while you isolate the actual production data and detail values to meet security rules, the structural results of this analysis (such as metadata and format) is shared with the DataStage and QualityStage users, with the unified InfoSphere Information Server repository.

Isolating engine nodes

As shown in Figure 5-6, an engine is dedicated to Information Analyzer on a separate server, which completely isolates the production data from the development environment, developers, and development data. In addition, built-in security features of InfoSphere Information Server, as described in “User authentication and roles” on page 183, ensure that users derive authority from the union of their product module roles in InfoSphere Information Server and the projects in the product modules with which they are entitled to work. However, you can additionally increase security by isolating engine nodes and IADB database from the DataStage or QualityStage environment at the file system level also.

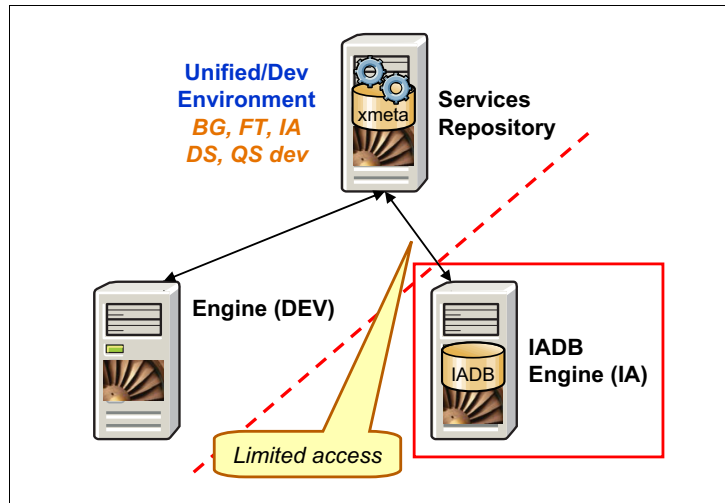


Figure 5-6 IA engine with IADB on stand-alone server

Before you analyze the production data, at certain times you might need an IA environment with non-production data. This source data can be created by promoting non-confidential data from the production environment or masking the confidential data depending on your need. With Information Analyzer, you can specify a different IADB by project basis, so by creating the IADB on different

nodes or file systems with appropriate permissions, the environments with different types of data can be isolated.

Figure 5-7 is an example of specifying IADB in an Information Analyzer project.

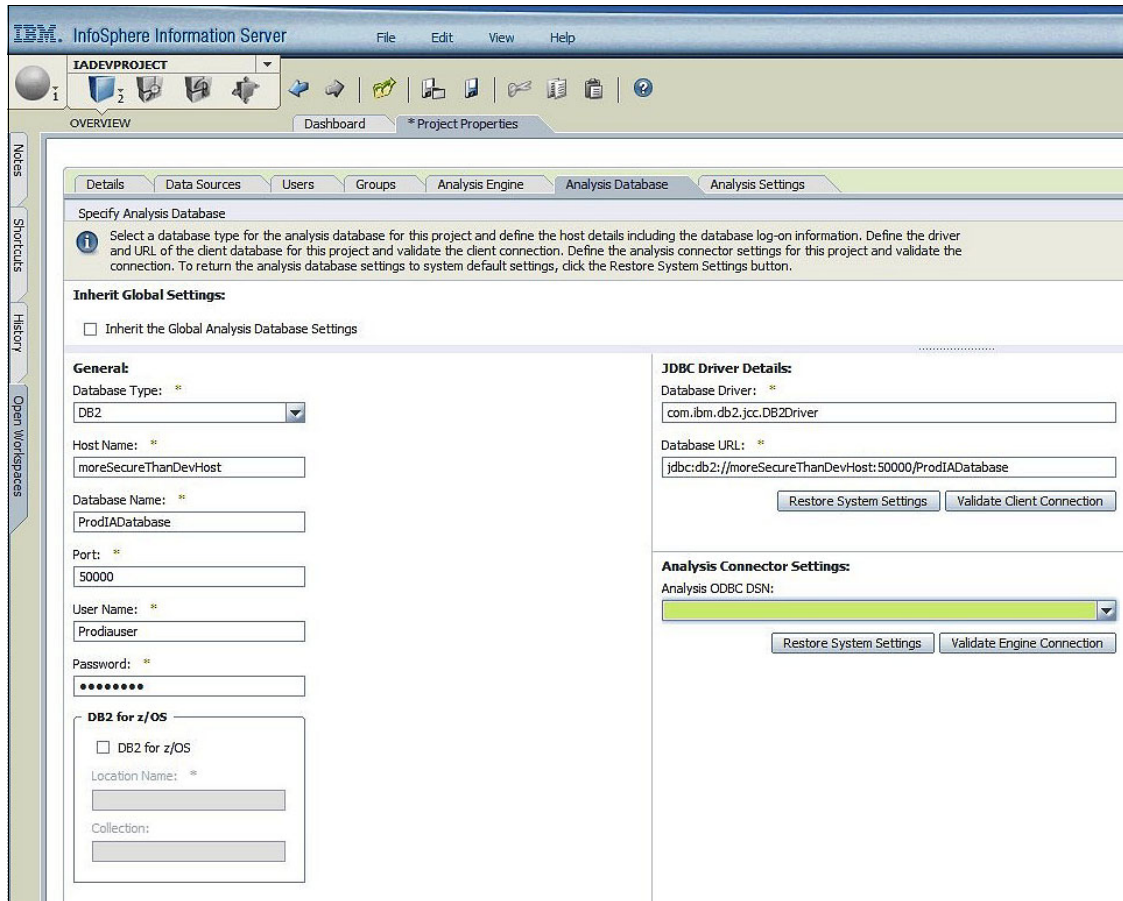


Figure 5-7 Specifying IADB on a project basis

User authentication and roles

InfoSphere Information Server supports role-based access control. Users derive authority from the union of their roles in InfoSphere Information Server, their roles in the suite component, such as DataStage and Information Analyzer, and the projects that they work with.

Figure 5-8 shows an overview of InfoSphere Information Server security roles.

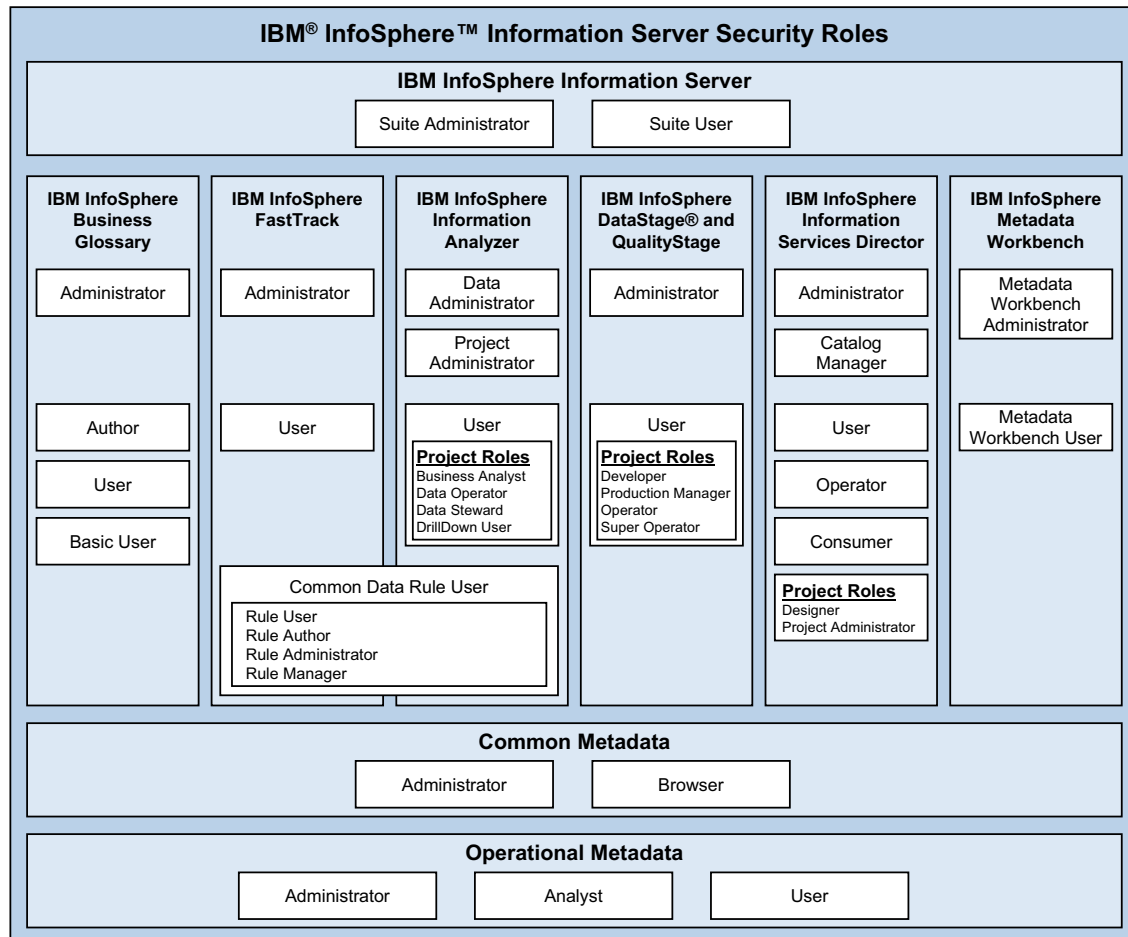


Figure 5-8 IA Server security roles

By creating separate users with separate roles for Information Analyzer and DataStage and QualityStage, you can control the access to each product. Authentication can also be set at the project level, so access can be controlled when multiple projects exist in Information Analyzer. Details about user roles are explained in *IBM InfoSphere Information Server Administration Guide*, SC18-9929-05, which is at the following information center:

http://publib.boulder.ibm.com/infocenter/iisinfsv/v8r7/topic/com.ibm.swg.im.iis.found.admin.common.doc/topics/c_wdisad_Roles.html

5.3 REPORTING architecture

REPORTING architecture has a dedicated metadata environment to provide all stakeholders full-time access to the various types of metadata (business and technical) that exist within InfoSphere Information Server.

In this dedicated REPORTING environment, regardless of the environment in which the metadata is created and persisted, the relevant metadata is copied to this separate instance of InfoSphere Information Server that is dedicated to accumulating metadata.

With this architecture, the system can handle a massive number of user requests, such as viewing Business Glossary, processing Data Quality rules or generating Metadata Workbench reports, without affecting the other existing environments.

Deciding which architecture to choose

If you are using Information Analyzer, FastTrack, Business Glossary, and Metadata Workbench with DataStage, QualityStage, and Information Services Director, but the UNIFIED/DEV environment is much affected and cannot support your requirement, consider choosing this REPORTING architecture. The requirement includes having a large number of users (for example, 100 or more) using Metadata Workbench or Business Glossary, or your glossary is intricate.

Typical structure

As shown in Figure 5-9 on page 186, the typical structure for this architecture is a UNIFIED/DEV, TEST, PROD, REPORTING, four-environment structure.

Depending on the product modules being used, and the collaboration among them, in some cases UNIFIED/DEV might be only a simple DEV environment. For example, if the Information Analyzer Data Quality rules function is used, and business analysts review the results to report to business, then they are not reusing the metadata that is generated by Information Analyzer for any other

purpose. In this case, there is no collaboration with other product modules so you might not need a UNIFIED environment, therefore the environment might be a plain DEV environment.

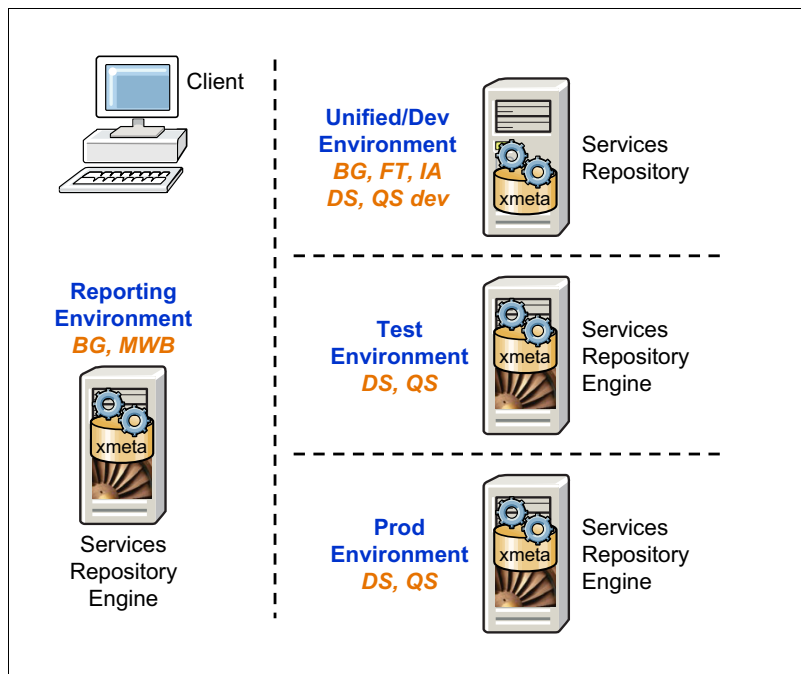


Figure 5-9 Typical architectural structure

Topology

For the UNIFIED DEV or DEV environment, a three-tier configuration is typical. The reason is because compiling and testing of many concurrent users can affect the performance of job developers.

For TEST and PROD environments, most clients use a standard two-tier configuration for security, usage, and availability considerations.

For REPORTING environments, a two-tier configuration is also typical. But, depending on the reporting requirement, a three-tier configuration would also be appropriate, because you can scale the services tier as described in Chapter 2, “InfoSphere Information Server installation topologies” on page 19. Also, if you choose to add another engine to an existing environment (DEV, TEST, PROD), this case will also be a three-tier configuration.

Implementation

Figure 5-10 on page 189 shows the implementation image. The basic implementation steps are as follows:

- ▶ DataStage and QualityStage jobs follow the typical DEV-TEST-PROD development lifecycle model.
- ▶ For metadata reporting purposes, you must copy DataStage and QualityStage job design and operational metadata, and import external metadata into the REPORTING environment.
 - DataStage/QualityStage job-design metadata is generated in the UNIFIED/DEV environment. Therefore, you must copy this metadata to the REPORTING environment.

Design: FastTrack mappings and Information Analyzer Rules Stages that are used in DataStage jobs is a part of this job design. This step can be done by any of the tools explained in Chapter 3, “Metadata management” on page 61.

- DataStage or QualityStage operational metadata is generated to an XML file that is stored on the file system at job run time. This operational metadata can be imported into xmeta through a shell script (`RunImportStart.sh`) or a batchfile (`RunImportStart.bat`) that is located on the engine tier of the PROD environment. Because the credential and connection for loading the metadata into the target xmeta is specified in a properties file (`runimport.cfg` or `runimport.cfg.unix`), you must specify the REPORTING environment instead of PROD environment for the target xmeta, and the operational metadata will be loaded into the REPORTING environment. Details about managing the operational metadata is described in *IBM InfoSphere Information Server, Guide to Managing Operational Metadata*, which is at the following address:

<http://publibfp.boulder.ibm.com/epubs/pdf/c1934770.pdf>

Values: Operational metadata is needed when your design metadata is insufficient for your metadata reporting. This situation can often happen when you use parameters without appropriate default values in your jobs. Operational metadata has the values used in execution, so you will not need to stitch the metadata manually (see Chapter 3, “Metadata management” on page 61).

- If you have a requirement to include external assets such as PL/SQL, and COBOL programs in your metadata reporting, you must also import these external metadata into the REPORTING InfoSphere Information Server repository. This can be done by multiple ways, described in Chapter 3, “Metadata management” on page 61.
- ▶ You must copy the business metadata to REPORTING from the UNIFIED/DEV environment.

Business metadata is generated by the Business Glossary authors in the UNIFIED/DEV environment but is consumed by the Business Glossary users in the REPORTING environment. Therefore, you must import the glossary into the REPORTING environment.

Business Glossary: If the Business Glossary is not used in collaboration with other product modules, the glossary can also be both generated and consumed in the REPORTING environment.

Business Glossary has its own development lifecycle function, which does not follow the typical DEV-TEST-PROD development lifecycle as do DataStage and QualityStage.

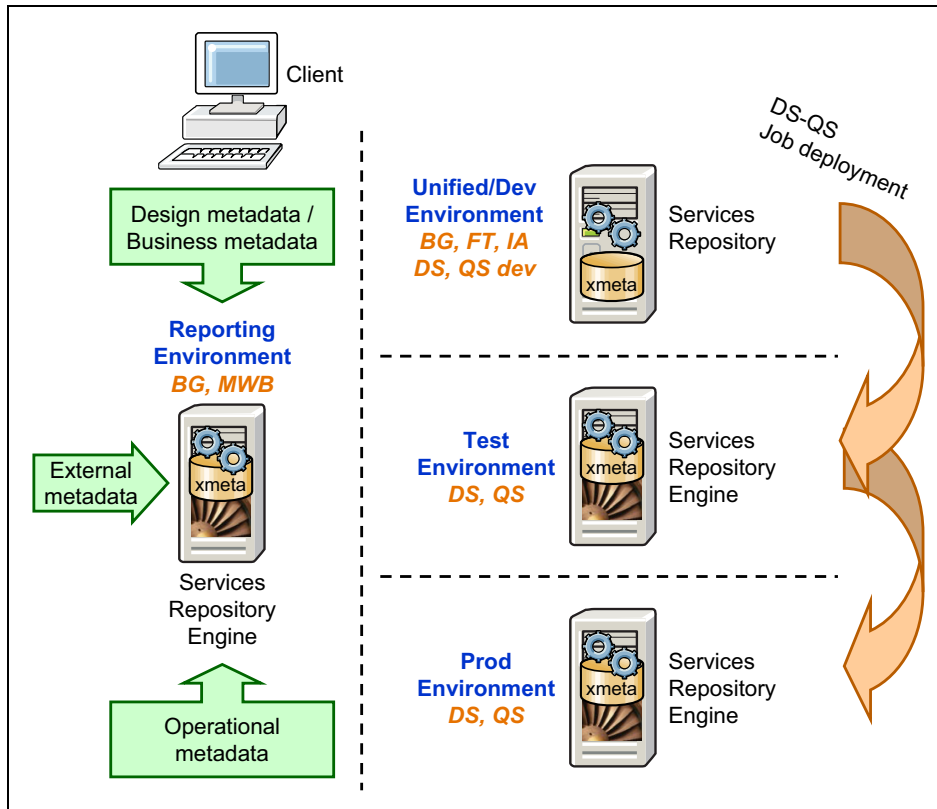


Figure 5-10 Implementation image

Advantages and disadvantages

There are advantages and disadvantages for this REPORTING architecture.

Advantages

The following advantages apply:

- ▶ The PROD-TEST-DEV environment is not affected by the various metadata needs, because metadata reporting is done on a dedicated REPORTING environment.
- ▶ The DEV environment is minimally affected by the Information Analyzer profiling of data because Information Analyzer has a different engine.
- ▶ Production data is secure by separating the Information Analyzer engine and IADB from the DEV environment.

Disadvantages

The following disadvantages apply:

- ▶ For operational metadata, the properties files might require slight modification to account for the fact that the metadata was generated in a separate environment.

Change management process: A well-understood change management process must be in place for such an environment.

- ▶ Additional effort is required to copy the metadata to the REPORTING environment.
- ▶ Additional hardware and software are required for this reporting environment.

Options: Although jobs do not run on the reporting environment, the DataStage engine is required to load job designs. Reporting does not affect the running of the jobs, so adding another engine to any of the existing environments (DEV, TEST, PROD) might be an option because of license considerations. In this case, consider the impact on the service-intensive processes, such as job compilation. In terms of using the operational data, installation of the installation tag (itag) in the PROD environment has an advantage: changing the host name for the metadata reporting is not necessary.

5.4 GOVERNANCE architecture

If you are not using DataStage or QualityStage, but want to use InfoSphere Information Server metadata reporting or governance functionality, you can create this stand alone GOVERNANCE environment.

In this case, external metadata can be analyzed and reported, and an enterprise glossary can be created or shared to increase governance in your company.

In some cases, when you are using Information Analyzer or FastTrack without collaboration with other modules this environment is also proper. Example cases are explained in “Deciding which architecture to choose” on page 191.

Deciding which architecture to choose

The main products that are used in this environment are Metadata Workbench and Business Glossary.

Depending on the functions you are using and whether there is collaboration between multiple product modules, having Information Analyzer and FastTrack in this environment are also appropriate. If you also use DataStage or QualityStage, one of the other three architectures described in this chapter should be more appropriate for your case.

In the following cases, functions of Information Analyzer and FastTrack are used as a single product without collaboration with other product modules. Other cases also exist.

- ▶ The profiling function of Information Analyzer is used for data quality assessment. In this case, business analysts review results and report back to business, source system owners, stewards, and so forth, but they are not reusing metadata for any other purpose; and, there is no collaboration with other product modules.
- ▶ Data quality rules functions of Information Analyzer are used through the Information Analyzer GUI (not the rules stage in DataStage).
- ▶ Mapping specification function of FastTrack is used for governance, but not for creating DataStage job templates.

Typical structure

As shown in Figure 5-11, the typical structure for this architecture is a stand-alone GOVERNANCE environment.

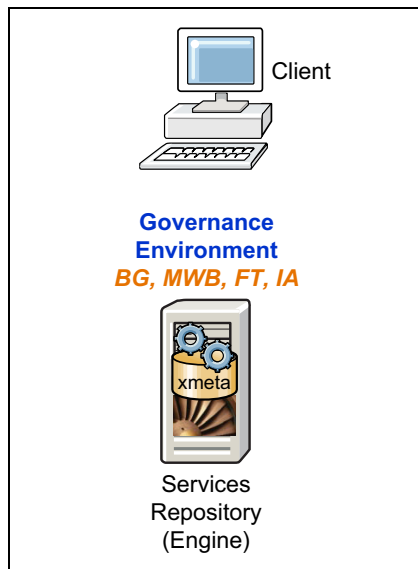


Figure 5-11 Structure for stand alone governance

Engine: The engine is not required, unless you are using Information Analyzer.

Topology

The common topology for the GOVERNANCE environment is a two-tier, stand-alone environment. Depending on the load, you may scale the environment. There are multiple patterns for scaling; details are described in Chapter 2, “InfoSphere Information Server installation topologies” on page 19.

Implementation

Figure 5-12 on page 193 shows the implementation image. The basic steps are as follows:

- ▶ Import external metadata by using the tools described in Chapter 3, “Metadata management” on page 61.
- ▶ Business metadata is generated and consumed in the Metadata Reporting environment, so you do not need to copy anything regarding the business metadata.

Lifecycles: As explained in Chapter 4, “Lifecycle management of assets” on page 75, Business Glossary has its own development lifecycle function, which does not follow the typical DEV-TEST-PROD development lifecycle as do DataStage and QualityStage.

- ▶ FastTrack mapping specifications are also generated and consumed in the Metadata Reporting environment, so you do not need to copy anything regarding the mappings.

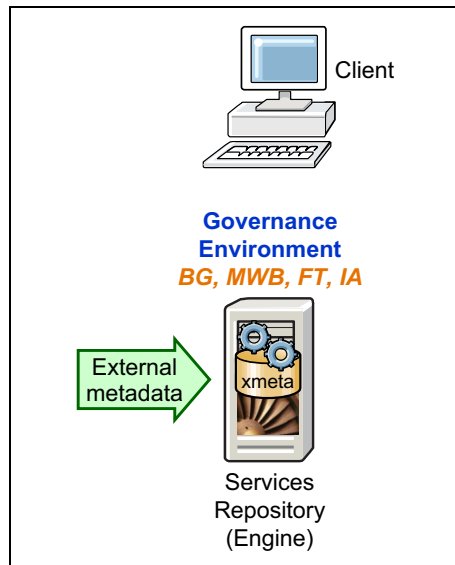


Figure 5-12 Governance implementation image

5.5 Choosing which architecture to deploy

This chapter introduces four deployment architectures, and considerations for those architectures. So, how do you choose which architectures to deploy for your particular situation?

The main determining factors for deciding the architecture are as follows:

- ▶ Which products are you using?
- ▶ Do you have metadata collaboration (reuse of metadata for other purposes) between multiple products?
- ▶ How many users will do lineage, impact analysis, and use the glossary?

- ▶ How intricate is your glossary?
- ▶ What is your environment capacity?

The chart in Figure 5-13 can help you choose the appropriate architecture for your situation.

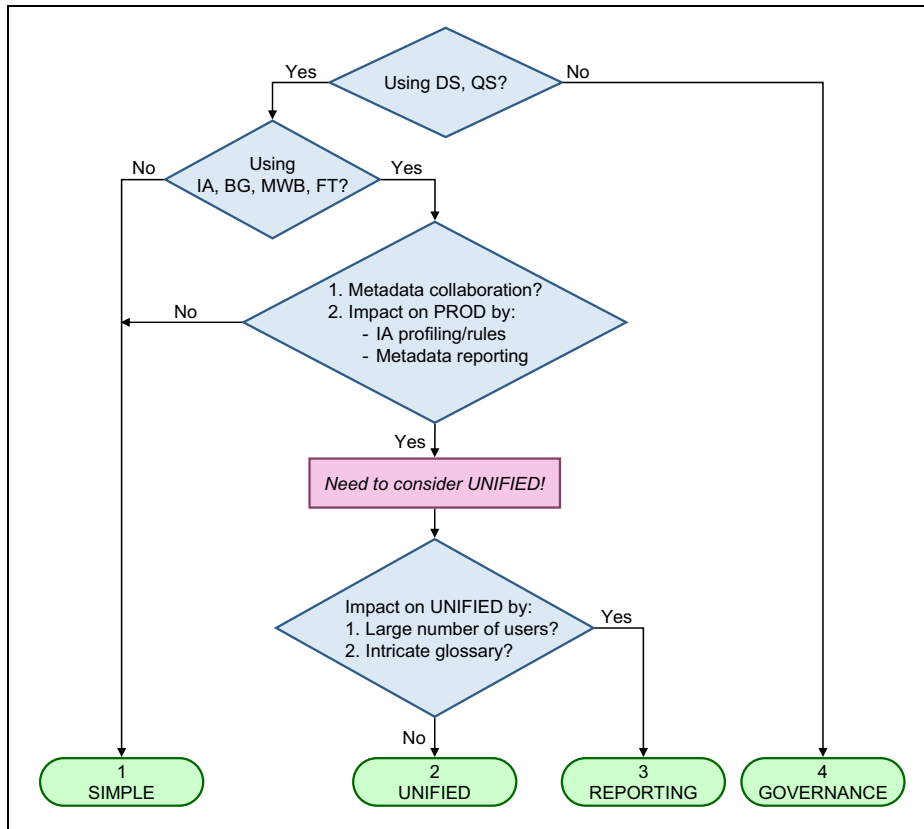


Figure 5-13 Architecture choice decision flow chart



Deployment use cases

The beginning of this book provides an overview of InfoSphere Information Server, InfoSphere Information Server functionality of product modules, infrastructure of server components, architectural topologies, and deployment of InfoSphere Information Server. It describes the administrative management of software that is developed by using relevant InfoSphere Information Server product modules, the software lifecycle, and the various environments that support the software deployment phases through delivery of the final product, which is the ultimate objective. Another objective is to provide additional information about those concepts to help promote a common understanding of them.

An important concept to understand is that multiple “final products” can result from a data integration program that takes advantage of InfoSphere Information Server. The typical final product is the *data*. This data product might take the form of reorganized, cleansed, and enriched content that can be used by applications for business intelligence, master data management, and data warehousing, as examples.

However, when using InfoSphere Information Server, an additional final product results from the *metadata*. This metadata product is more than a by-product of the ETL processes and can itself serve many purposes. It can provide business meaning to the data, and governance over business and information technology (IT) processes and information. It also provides reports on the impact that this data has on applications, people who need this information, the processes

involved in creating and delivering this information, and validation of the data flow and processing that delivers it from source to target.

Some customers use only the data as their final product; others use only the metadata product; and more often, others use both data and metadata final products. These distinctive “final products” have different development and lifecycles, and therefore, provide the impetus for several deployment architectures described in this book.

This chapter reviews three scenarios of differing business requirements for implementing solutions based on InfoSphere Information Server. The chapter builds upon the principles that are described in the previous chapters to design an appropriate deployment architecture, which can provide a successful implementation of programs such as data integration, data governance, and master data management. This example-based design analysis can help you successfully use these same principles for designing an appropriate InfoSphere Information Server deployment architecture, in accordance with your specific business requirements to be implemented, and final products to be delivered.

6.1 Metadata differentiator

The previous chapters in this book describe InfoSphere Information Server in the context of its infrastructure, the functionality of its product modules and where the product modules fit within the infrastructure, including the manner in which the InfoSphere Information Server product modules integrate with one another. Much of their integration is based on the accumulation of both internal metadata that is generated by the product modules themselves, and external metadata that is imported into the shared InfoSphere Information Server repository by using various InfoSphere Information Server tools. This metadata component is usually a significant differentiator from traditional data integration processes, and requires a new understanding for using it in an optimal fashion. In addition, the chapters presented a number of deployment options to demonstrate how InfoSphere Information Server supports a vast number of complex business requirements.

This shared metadata is not administered or maintained in the same manner as is developed software code, such as DataStage job design code. For example, deploying the Business Glossary (BG) product module in more than one environment is generally not necessary because the Business Glossary has its own built-in work flow and does not require a typical DEV-TEST-PROD software development cycle. Therefore, the metadata that is generated by BG is persisted only in the environment in which BG is deployed. However, if the BG is deployed in a UNIFIED DEV environment (as described in Chapter 5, “Metadata

deployment architectures” on page 169), it might require production-level common technical metadata for asset assignment purposes, rather than the development-level assets that are used by other product modules that are deployed in the same environment. In this example, both development-level and production-level metadata can exist in the same environment.

In addition, the lifecycle and life expectancy for code that is developed for a particular data integration project does not necessarily synchronize with the metadata lifecycle. Finally, the reporting that results from *metadata* does not necessarily serve the same purpose as the reporting that results from the *data*, which came from the data integration development processes. These issues are what leads to the varied deployment architectures that are analyzed in this chapter.

6.2 Sample use case scenarios

Three use cases are presented in this section along with their business requirements. The task is to analyze the requirements for each scenario and apply the principles that are presented in this book to determine an appropriate deployment architecture to meet the business objectives. The first scenario is relatively simple and modest in terms of users and resources that are required. The second scenario is a bit more complex. The final scenario, is the most complex in terms of both data and metadata reporting requirements, and focuses on “how to make it work.” A possibility is that multiple architectures can fulfill the business requirements, however, only one is presented here.

The following process assesses the architecture and deployment environments:

1. Use the decision chart in Figure 5-13 on page 194 to determine an appropriate deployment architecture (how many, and what kinds of environments are required to support the product mix).
2. Determine in which environments each of the product modules should be deployed.
3. Evaluate requirements for high availability.
4. Design a topology that is based on business and IT requirements, availability and product mix for each environment.

Before you assess the architecture and deployment for InfoSphere Information Server, you must determine the hardware resources (CPU cores, RAM, disk volumes and types, and so forth) that are required to support the expected load on the hardware in each environment, although sizing is not in scope for this exercise. After you determine how to deploy all of the product modules and the infrastructure to support them, establish a process to ensure that all repository

assets that are required for each product module are available in the environment in which each product module is deployed. The final step of each scenario in the process demonstrates how to “make it (the implementation) work.”

6.2.1 Scenario 1: Simple

A small to medium sized industrial company recently purchased InfoSphere Information Server and is planning to use both DataStage and QualityStage for data integration. In addition, the company purchased Business Glossary, but the company does not intend to roll it out to the entire enterprise at this time. The company also purchased Metadata Workbench (MWB) and intends to use it for the data lineage capability.

The company is allocating three or four developers for the duration of the initial phase of the project and will add more developers when more business units agree to participate in the data integration program. The current production requirements dictate that the company should have a high availability failover environment if the primary production server fails. After reviewing the various types of available options, the company determined that a simple active-passive configuration can meet the company needs. The company discussed the benefits of having the development environment be clustered to allow continuous development efforts, but could not justify the additional hardware costs. The only environment that will be configured for high availability will be the production environment.

Which deployment architecture to use

The first step in determining the deployment landscape is to review the decision chart (Figure 5-13 on page 194) to determine an appropriate deployment architecture. Answers to decision-chart questions for this scenario are as follows:

1. Using DS, QS?

Yes

2. Using IA, BG, MWB, FT?

Yes

Key decision point: Next is the key decision point that determines whether the SIMPLE (development-test-production) deployment architecture is sufficient, or the scenario requires the advanced design features that are typical of the UNIFIED environment.

3. Will related workload for metadata collaboration require additional resource for IA profiling and data rules or metadata reporting?

No

Explanation: At this time, because the use of Business Glossary is limited (limited to a few business units), and only data lineage reports are generated from the Metadata Workbench, it does not appear that there will be enough collaboration to justify a UNIFIED development environment. Furthermore, there does not seem to be a significant enough load expected on the production environment that would necessitate a separate metadata reporting environment. (Production is a natural environment for deploying these product modules.)

The result from using this decision tree is to suggest a SIMPLE deployment architecture that will consist of development, test, and production environments.

Where to deploy the product modules

The product modules can be deployed to the following environments:

- ▶ DataStage and QualityStage will be deployed in all environments according to standard software development lifecycle (SDLC) methodology.
- ▶ Business Glossary will be deployed only in the production environment, for accessibility by the authorized users. Glossary authorship will be handled through the built-in work flow mechanisms, preventing unauthorized users from viewing pre-approved content.
- ▶ Metadata Workbench will definitely be deployed in the production environment, however, there might be justification to deploy Metadata Workbench in the development environment also. More information regarding the requirements of the developers is needed to determine this deployment.

Availability requirements

According to the business requirements for the industrial company in the example, the company requires active-passive high availability in the production environment alone. All other issues being equal, and in consideration of simplifying the high availability configuration, a suggestion is to install all three server tiers on the same server (physical or virtual) for the production environment.

Figure 6-1 shows the SIMPLE deployment based on a single server that hosts all three InfoSphere Information Server tiers (services, repository, and engine). This particular configuration is appropriate for both the development and test environments to support the use cases of the medium-sized industrial company in this scenario.

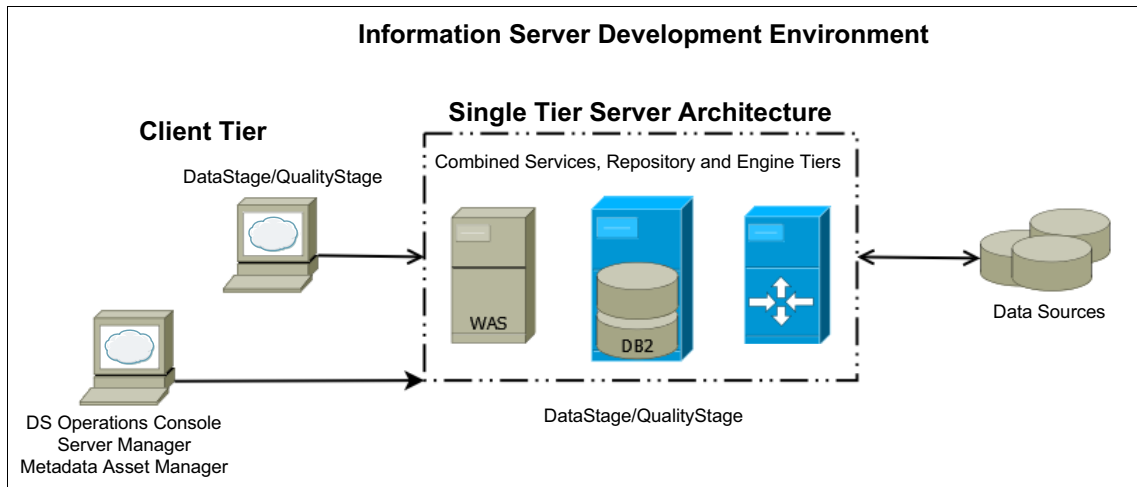


Figure 6-1 SIMPLE deployment architecture for development and test environments

The production environment requires high availability in the form of active-passive, to ensure continuous flow of data to its destinations. Figure 6-2 on page 201 shows the configuration in both active mode and after a fail over to the secondary server. Notice the front-end HA manager that monitors the “health” of the active server and determines when failing over is necessary. The HA manager can be implemented by a hardware device that is designed for this purpose, or by software or scripts that are developed to monitor key processes and events that dictate the need to fail over to the passive server, and execute on the failover process. For more information about active-passive availability architecture, see 2.7, “InfoSphere Information Server clustered installations” on page 28.

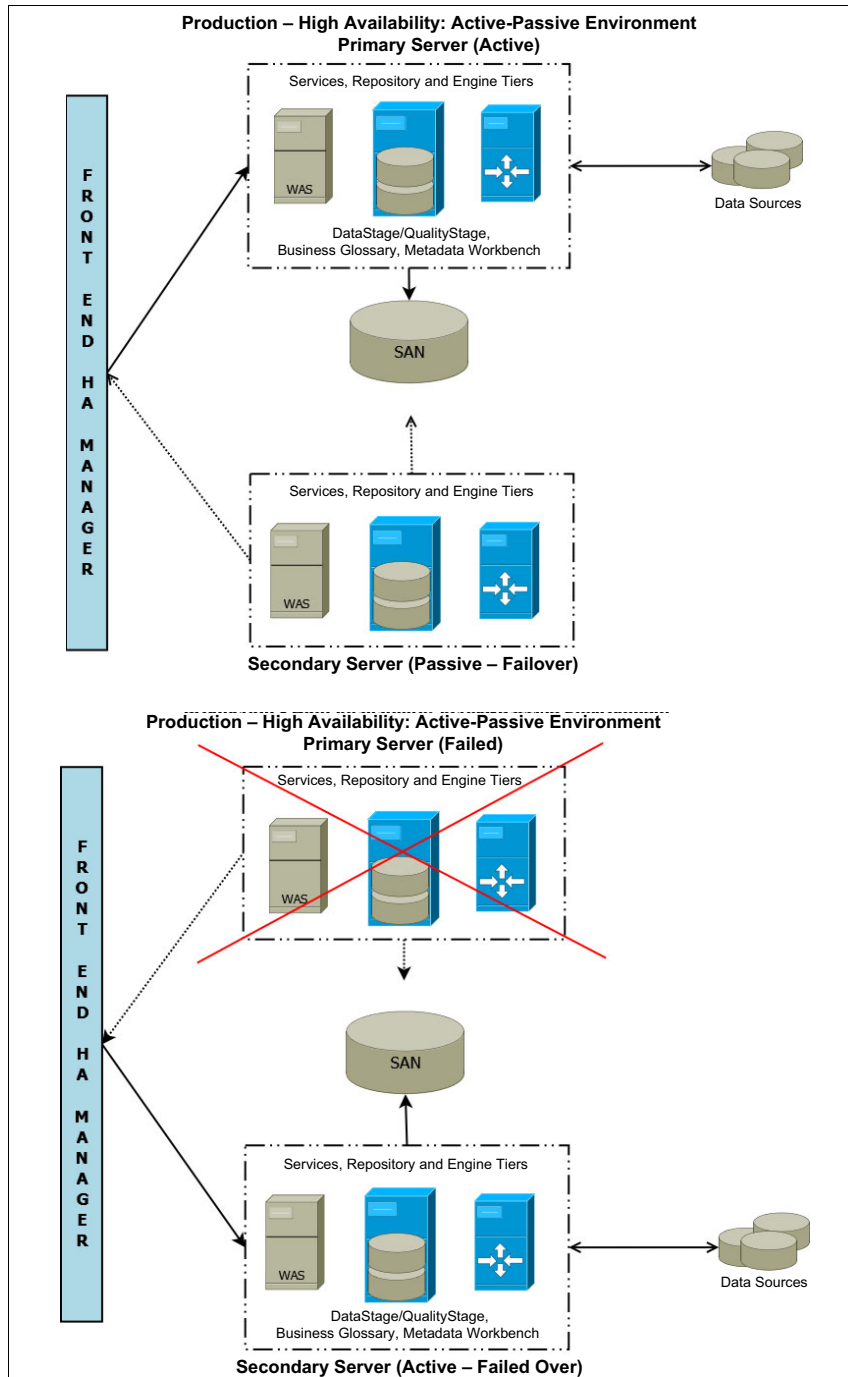


Figure 6-2 SIMPLE high availability configuration for production environment

How to make it work

Chapter 4, “Lifecycle management of assets” on page 75 describes the SDLC deployment process in sufficient detail. It describes how DataStage code can be deployed through the environments from development, to test, to production. Because deploying DataStage jobs to the production environment is part of the SDLC process, an assumption is that all DataStage job design metadata will be available for both BG and MWB in the production environment, as needed. Therefore, it is not included in this process.

In this scenario, Business Glossary creates its own metadata and persists it in the repository of the production environment, where BG is deployed. Furthermore, BG uses other metadata assets that are already persisted in the production environment (for example, common technical metadata assets such as databases and data files that are used by other product modules in the production environment) repository. Because BG does not require any assets that only exist in other environments, there is no need to deploy any other assets from other environments to production for the benefit of BG.

There might be some external technical metadata that BG can use. This needs to be identified, imported into the repository, and assigned to BG terms as necessary. A typical example of external technical metadata that is useful to BG is business intelligence metadata. However, in this case, importing external metadata is not part of the deployment process, rather it is part of the general usage of the InfoSphere Information Server product modules. Furthermore, an often suggested step is to import the external technical metadata into each environment in which it is required, rather than to deploy from one environment to another. However, in this instance, the external business intelligence metadata, if relevant, will be imported only into the production environment (for use with BG and MWB).

The business requirement for data lineage means that the Metadata Workbench requires data sources and targets in the repository, possibly business intelligence metadata, and DataStage job design. Because the DataStage job design is deployed as part of the DataStage deployment process, it can be assumed that the necessary jobs are already present in the repository for Metadata Workbench to use. However, certain tips and tricks can ensure data lineage, and minimize the manual intervention that is at times required.

Details about the basic requirements, prerequisites, and suggested practices for generating proper data lineage reports is in *Metadata Management with IBM InfoSphere Information Server*, SG24-7939. One point that is important to mention is that DataStage job design often relies on using job parameters as stage parameter values, where the values are not “known” to InfoSphere Information Server until run time. Because Metadata Workbench relies on DataStage job design in generating lineage reports, the use of variables in the

job design might mean that Metadata Workbench is missing key information that is needed for the lineage reports. One solution is to generate, collect, and load operational metadata (OMD) from the DataStage job runtime execution. In this manner, the variable values are instantiated and, therefore, can be used by Metadata Workbench to supplement the DataStage job design metadata. Other solutions are not related to the deployment of metadata assets, and are therefore, out of scope for this discussion.

Operational metadata is collected in the DataStage runtime file system in XML structured files; see Figure 6-3. A shell script (Figure 6-4 on page 204) reads the operational metadata files and loads the content into the InfoSphere Information Server repository as operational metadata, thereby instantiating parameterized values among other items. Because Metadata Workbench is deployed in the production environment, and the operational metadata is generated by executing DataStage jobs in the production environment, deploying operational metadata from a separate environment into the production environment in this scenario is not necessary.

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <Run StartedAt="2012-07-23T16:45:25" CreationModel="DataStage8" Message="Fi
3  <Deployment>
4  <SoftwareResourceLocator>
5  <LocatorComponent Class="Computer" Name="orion" />
6  <LocatorComponent Class="SoftwareProduct" Name="DataStage" />
7  <LocatorComponent Class="SoftwareGroup" SubClass="Project" Name="zeta" />
8  <LocatorComponent Class="SoftwareExecutable" SubClass="Job" Name="JobDailyR
9  </SoftwareResourceLocator>
10 </Deployment>
11 <Design>
12 <SoftwareResourceLocator>
13 <LocatorComponent Class="Computer" Name="orion" />
14 <LocatorComponent Class="SoftwareProduct" Name="DataStage" />
15 <LocatorComponent Class="SoftwareGroup" SubClass="Project" Name="zeta" />
16 <LocatorComponent Class="SoftwareDesign" SubClass="Job" Name="JobDailyRunIn
17 </SoftwareResourceLocator>
18 </Design>
19 <ActualParameters>
65 </ActualParameters>
66 <Events>
67 <Event Type="Read" StartedAt="2012-07-23T16:45:27" FinishedAt="2012-07-23T1
68 <DataResourceLocator>
69 <LocatorComponent Class="Computer" Name="orion" />
70 <LocatorComponent Class="SoftwareProduct" Name="DB2" />
71 <LocatorComponent Class="DataStore" SubClass="Database" Name="SAMPLE" />
72 <LocatorComponent Class="DataSchema" SubClass="Schema" Name="TST_CTRL" />
73 <LocatorComponent Class="DataCollection" SubClass="Table" Name="POL" />
74 </DataResourceLocator>

```

Figure 6-3 Sample operational metadata (OMD) XML file

```

1  #!/bin/sh
2  case "$0" in
3      *) cmd="$0" ;;
4      *) cmd=`which "$0"` ;;
5  esac
6  dir=`dirname "$cmd"`
7  nodebin="$dir/."
8  . "$nodebin/setupEnv.sh"
9  CLASSPATH=""
10 CLASSPATH="${CLASSPATH}."
11 CLASSPATH="${CLASSPATH}./conf:"
12 CLASSPATH="${CLASSPATH}./lib/java/xmeta_cache.jar:"
13 CLASSPATH="${CLASSPATH}./lib/java/framework.jar:"
14 CLASSPATH="${CLASSPATH}./lib/java/repository.jar:"
15 CLASSPATH="${CLASSPATH}./lib/java/log4j-1.2.8.jar:"
16 CLASSPATH="${CLASSPATH}./lib/java/metadata_services-client.jar:"
17 CLASSPATH="${CLASSPATH}./lib/java/metadata_services-asb.jar:"
18 CLASSPATH="${CLASSPATH}./lib/java/OMDModel_api_gen_ecore.jar:"
19 CLASSPATH="${CLASSPATH}./lib/java/models_xmeta_api.jar:"
20 CLASSPATH="${CLASSPATH}./lib/java/ecore-2.2.2.jar:"
21 CLASSPATH="${CLASSPATH}./lib/java/common-2.2.2.jar:"
22 CLASSPATH="${CLASSPATH}./lib/java/ecore.change-2.2.2.jar:"
23 CLASSPATH="${CLASSPATH}./lib/java/ecore.xmi-2.2.2.jar:"
24 CLASSPATH="${CLASSPATH}./lib/java/ecore.sdo-2.2.2.jar:"
25 CLASSPATH="${CLASSPATH}./lib/java/commonj.sdo-2.2.2.jar:"
26 CLASSPATH="${CLASSPATH}./lib/java/runimporter.jar:"
27 CLASSPATH="${CLASSPATH}${ISF_CP}:${J2EE_CP}"
28 ../apps/jre/bin/java ${J2EE_OPTS} -classpath "${CLASSPATH}" com

```

Figure 6-4 RunImportStart.sh shell script for loading OMD into InfoSphere Information Server repository

6.2.2 Scenario 2: Mixed

A large bank that uses many of the IBM Information Management products has used DataStage for many years. The bank has never needed high availability for any environment until a recent merger with another large bank. The combined development staff consists of over 100 developers for mostly batch-type ETL processing.

The new plan is to use all of the product modules of InfoSphere Information Server (excluding Business Glossary), including Information Analyzer (IA), FastTrack (FT), DataStage (DS) and Metadata Workbench (MWB).

The production environment will not change much from its current configuration, except for the addition of high availability to ensure that data is available for reporting by 7:00 a.m. The UNIFIED collaborative development environment needs to provide high availability for a large number of concurrent users.

Which deployment architecture to use

We use the decision chart (Figure 5-13 on page 194) to determine the following responses:

1. Using DS and QS?

Yes

2. Using IA, BG, MWB, FT?

Yes (IA, MWB, FT)

3. Will related workload for metadata collaboration require additional resource for IA profiling and data rules or metadata reporting?

Yes (metadata collaboration primarily with FT, but also IA)

Explanation: As soon as a requirement exists for collaboration of this nature, this collaboration indicates that these product modules (IA and FT, in particular) should be deployed in the development environment where their metadata can be used by DataStage developers. However, IA should execute on production-level data (even if it is deployed in the development environment). As described in Chapter 5, “Metadata deployment architectures” on page 169, this problem can be resolved by configuring separate engines for the DataStage development work, and for the IA work. DataStage projects will execute on one engine, while IA profiling, rules execution, or both, run on a second engine, with no crossover.

The question remains whether there is also a need for a separate metadata Reporting environment. The response to the final question determines the answer.

4. Large number of users for metadata reports and queries, and an intricate glossary?

No (A large number of developers, but not for FT or IA, and metadata reports are generated from those product modules.)

The combination of product modules sharing metadata, but processing different data (development versus production), results in the need for the UNIFIED environment. However, because there is also ETL development work with DataStage, there will also be test and production environments, as fits with the development standards of the bank. Because few users will be placing a load on

the metadata reports (lineage, data rule results and queries), establishing an additional reporting or metadata only Reporting environment is not necessary at the present time.

Where to deploy the product modules

The product modules can be deployed to the following environments:

- ▶ DataStage and QualityStage will be deployed in all environments as dictated by standard software development lifecycle methodology.
- ▶ The FT product module will be deployed only in the UNIFIED environment because the FT mapping specifications are only required to assist the developers. If the FT metadata is required for lineage in a production environment, the FT assets can be exported from the UNIFIED environment and imported into the production (or reporting) environment by using the Asset Interchange functionality from the `istool` command-line interface (CLI). However, this step is not required at the present time.
- ▶ The IA product module will be deployed only in the UNIFIED environment and will use a separate engine from the engine that the DataStage developers will use. In this way, DataStage developers can access only development data and Information Analyzers can access only production data,
- ▶ Because collaboration is a central business requirement, MWB will also only be deployed in the UNIFIED environment, because almost all of the metadata that is required by MWB is persisted in the UNIFIED environment. Furthermore, although not enough justification exists to establish a metadata-only reporting environment, there likely will be sufficient lineage, user impact, and query reporting to place a greater load on the production environment. The UNIFIED environment topology will effectively handle these loads, as specified in the business requirements.

Availability requirements

The business requirements state that high availability should exist in both the UNIFIED and the production environments (and therefore, also the test environment, as a simulation for production). The high availability topology, suggested for the production system in the simple example (6.2.1, “Scenario 1: Simple” on page 198), is also appropriate for the test and production environments for this bank (see Figure 6-2 on page 201).

The big difference has to do with the UNIFIED environment. To support the large number of developers, and the Foundation Tools (particularly MWB), the suggestion is to cluster the WebSphere Application Servers in the services tier. This approach can provide the equivalent of an active-active high availability, but more important, it can support a large number of concurrent sessions, along with the web applications (namely MWB). In addition, the UNIFIED environment will

require two separate engines to support both DataStage development job executions, and also IA profiling, rules execution, or both.

Figure 6-5 shows the UNIFIED architectural diagram of this topology.

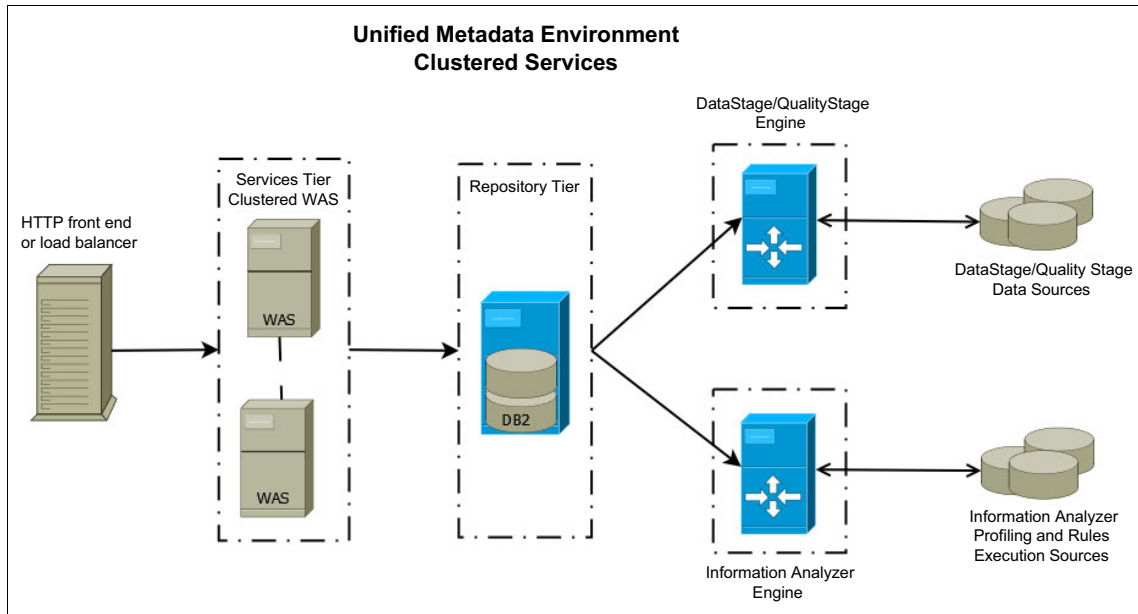


Figure 6-5 UNIFIED environment with clustered services tier

Note that the HTTP web server front-end, and possibly a load balancer also, can be installed on a separate server, or combined with one of the WebSphere Application Server cluster nodes.

How to make it work

The deployment architecture for this bank is fairly standard with the exception of the UNIFIED environment, which is unique to InfoSphere Information Server, with the inclusion of the InfoSphere Foundation Tools, such as IA, FT, and MWB. However, because all of the InfoSphere Information Server product modules that are involved in collaboration are deployed in the UNIFIED environment, all of the required assets should be available in the shared repository, because they are persisted by the deployed product modules themselves.

Persisted data: The InfoSphere Information Server repository does not persist any data from the enterprise data sources; only metadata is persisted from these sources. Therefore, although the repository is shared between the product modules that access various levels of data (one engine accessing development-level data and the other engine accessing production-level data), only the metadata from the various data sources is persisted in the repository, with controlled access only to the appropriate users.

The technical metadata that resides in the UNIFIED environment will include both the current production assets (because they were originally developed in the development environment) and the new development assets (that are currently being developed). This production-level metadata enables the connectivity for Information Analyzer to access and process the current production data sources (and their metadata). Furthermore, the DataStage jobs that are currently running in the production environment were first developed in the development (UNIFIED) environment. Therefore, these job designs should also be in the UNIFIED environment, available for Metadata Workbench for lineage and impact analysis reports.

The important point here is the total separation between DataStage (and DataStage developers) access to development-only data, and IA (and IA users) access only to production level data.

Development and production data can be separated in accordance with standard practices of many enterprises and international regulations by using the following steps:

1. Create DataStage projects for DataStage and IA in separate directories on the server (engine tier), each with their own respective engine.
2. Execute parallel jobs from DataStage only on the DataStage engine.
3. Execute parallel jobs from IA only on the IA engine.
4. Allow access to the DataStage directory only by appropriate DataStage users.
5. Allow access to the IA directory only by appropriate IA users.

However, FastTrack mapping specifications are likely using metadata that is not yet in production. These design specifications are the basis for new DataStage job development. Therefore this technical metadata is development level (until it is available in the production environment). Finally, the actual jobs that are being developed should be running against development-level data (and corresponding development-level metadata).

All of the development-level and production-level metadata is properly persisted in the repository and accessible to the product modules that are capable of using it. The only item missing at this stage is the operational metadata that is generated by DataStage and QualityStage jobs in the production environment to be used by Metadata Workbench for lineage and impact analysis. This metadata must be loaded into the repository in the UNIFIED environment and *not* in the production environment.

As described in 6.2.1, “Scenario 1: Simple” on page 198, the operational metadata is written to XML files in the DataStage and QualityStage file system by DataStage and QualityStage jobs that are executing in the production environment. So instead of running the shell script and loading the operational metadata into the production repository, a possibility is to first copy or move the XML files that contain operational metadata from the production file system into the file system that is accessible to the UNIFIED environment, Then run the shell script from the UNIFIED environment to load the operational metadata files into the UNIFIED environment repository.

Association: This step associates the operational metadata that is generated by the DataStage and QualityStage jobs that were executed in the production environment, and attaches them to the DataStage and QualityStage job designs that are persisted in the UNIFIED DEV environment.

6.2.3 Scenario 3: Complex

With the acceptance of the Solvency II Directive, a large multinational insurance company is beginning a new information governance initiative with support from the Corporate Executives and all lines of business. Although the company has well-documented processes that it follows for each new business project, the company does not have much experience in the area of governance. Therefore, the company purchased the complete Enterprise Edition of InfoSphere Information Server to help manage this initiative (along with IBM InfoSphere Guardium® for data security and other relevant IBM software). The company will begin with the design and development of a new data warehouse to support its business intelligence reporting with IBM Cognos. To get started, the company has included the IBM Insurance Information Warehouse (IIW) industry model for insurance that the company plans to customize to its needs with the InfoSphere Data Architect (DA) modelling tool. The IBM model includes a pre-populated Business Glossary that the company intends to use as a reference while the company slowly builds its own glossary to roll out to the enterprise in phases. In addition to the standardization of the company’s enterprise vocabulary, it will also use Business Glossary for capturing its business requirements for each new project, and standardizing these requirements where possible against its

enterprise glossary as the glossary relates to the company's business metrics, key performance indicators (KPIs), and general vocabulary.

The company is currently using spreadsheets to document source-to-target mapping specifications that the company uses for ETL design. The problems the company experiences with this mechanism are not being aware when a business analyst signs off on a document, or it requires revisions when there is overlap of requirements with other current projects, or whether the attributes were already mapped in another spreadsheet. All of these issues contribute to wasting valuable resources, and being error prone. The insurance company plans to replace its spreadsheets with FastTrack, to centralize the management of these mapping specifications, which includes mechanisms for browsing, searching, and "discovering" existing data sources for reuse. Initially, the company will use its physical design models (from DA) as a target for the ETL design until the design is finalized and instantiated in the Netezza® based data warehouse. Then, the company will automatically remap to the metadata of the implemented data warehouse on the Netezza platform.

From an ETL perspective, the company needs to migrate its existing data warehouse to the new model, and re-engineer the current ETL processes from the existing sources to the new warehouse. Unfortunately, the only documentation available to support the company's current ETL processes are the mapping spreadsheets (other than the code itself), which are not appropriate for this purpose. As part of the new initiative, the company will catalog its data sources for documentation purposes by importing the metadata into the InfoSphere Information Server shared repository. The developers will also need to understand the impact of the re-engineering work that they are executing, so they will use Metadata Workbench for the impact-analysis features.

The company will be using DataStage extensively for all of the new ETL processes that the company is currently implementing with another ETL tool. Additionally, DataStage will be able to use the data catalog from the shared repository, as a reference to the actual metadata from the data sources, which can be used for DataStage table definitions that are used in DataStage job designs. There will be several separate projects, and also a very large volume of data that will be processed daily within a relatively short time frame, because of the range of time zones across the enterprise and the relatively small window of time.

As part of the governance initiative, the insurance company needs to verify that the quality of its data fits within its data quality metrics. The company will be defining the quality rules and measuring adherence to these defined standards on a regular basis. On a further quality perspective, the company needs to verify that its regulatory reports for Solvency II are based on the data the company

thinks the Solvency II reports are. The auditors will check the data lineage of the regulatory reports by using Metadata Workbench, for inclusion as an audit trail.

Which deployment architecture to use

We use the decision chart (Figure 5-13 on page 194) to determine the following responses:

1. Using DS or QS (or ISD)?

Yes

2. Using IA, BG, MWB, FT?

Yes (all of them)

3. Will related workload for metadata collaboration require additional resource for IA profiling and data rules or metadata reporting?

Yes (significant collaboration and reporting)

4. Large number of users for metadata reports and queries, and an intricate glossary?

Yes (regular quality metric reporting, metadata impact analysis and data lineage reports, along with a full reference glossary and a governed glossary built from the bottom up, including assigned assets and other relationships; will be rolled out to the enterprise, so the user population will increase with time)

Explanation: A very large volume of metadata will be persisted in the repository, along with a significant number of relationships, between terms and various technical assets (data sources, targets and BI assets). This persistence means that a heavy load will be placed on the services tier and potentially the repository tier (although RDBMSs can generally support large volumes of data without requiring clustering). The impact of the metadata reporting is likely to be too much for either a production environment or a UNIFIED environment.

Given both the level of collaboration (DA and FT) and the metadata reporting requirements (IA data rules metrics, Business Glossary governance and Metadata Workbench impact analysis and data lineage), this insurance company should establish development, test, and production environments for the DataStage SDLC, and a separate reporting environment for the metadata reports and data quality metrics. Potentially, the development environment can be a UNIFIED environment for collaboration. However, because IA is not being used for profiling data, the collaborative assets can be produced in the reporting environment and deployed to development (if necessary), rather than deploying the relevant product modules in multiple environments.

Furthermore, collaboration with FT might also indicate the need for a UNIFIED environment. However, the best solution is for FT to be deployed with Business Glossary. Furthermore, the actual mapping specifications do not have to be in the same environment as DataStage (unless FT will be used to generate DataStage template jobs, which is not the use case here).

There are potentially multiple solutions to achieve the optimal deployment architecture for this (or any) customer. However, one of the core principles in determining a deployment architecture is to keep it as simple as possible. In the case of this insurance company, minimizing the number of environments in which a product module is deployed is helpful, as is minimizing the deployment of repository assets (business and technical metadata) from one environment to another. There should also be two distinct deployment architectures: DEV-TEST-PROD for the ETL processing, and reporting for the metadata reporting. Which artifacts will need to be shared between the two deployment architectures can then be determined.

Where to deploy the product modules

The product modules can be deployed to the following environments:

- ▶ DataStage (and QualityStage if required) will be deployed in the three standard environments (development, test and production) according to standard software development lifecycle (SDLC) methodology. In addition, they must be installed in the reporting environment to import DataStage jobs that will be used by MWB in formulating the data lineage reports.
- ▶ Information Analyzer will be used for the data rules execution and data quality reporting of these metrics (on production level data). Therefore, it belongs in the reporting environment, because this functionality is not used for collaborative purposes, but rather for reporting only.

Consider this situation: If DataStage developers are also using these data rules in the DataStage data rules stage, there is collaboration and a need to deploy IA in the development environment in addition to, or instead of, the reporting environment. Similarly, if IA is being used for profiling, then that is another collaborative function, which can justify its deployment in the development (UNIFIED) environment. However, this situation is not the case with this insurance company; IA will be deployed only in the reporting environment.

- ▶ Business Glossary has its own built-in work flow, which means it requires deployment in only one environment. Under these circumstances, Business Glossary will be deployed in the reporting environment, because its main purpose is for users to search, browse, and query the glossary as needed. However, this approach means that all metadata that Business Glossary needs to relate to must be persisted (or a copy persisted) in the reporting environment.

- ▶ Although FastTrack would be at home in the development (or UNIFIED) environment for collaboration purposes, it is best deployed in the same environment as Business Glossary to use the term-to-database column relationships for “automating” the “discovery” of source-to-target mappings. Therefore, FT will be deployed in the reporting environment (with the option of deploying its metadata to the development environment if needed).
- ▶ Metadata Workbench will be used both for the data lineage governance reports and for impact analysis for the developers. That requires Metadata Workbench to be deployed both in the development environment (for impact analysis on new sources that are not yet in production) and in the reporting environment (for data lineage against production assets). As an alternative, it is possible to deploy Metadata Workbench only in the reporting environment, but then it will be necessary to deploy too many technical assets from the development to the reporting environment, and that are not required by any of the other product modules in the reporting environment.

In summary, DataStage and QualityStage are deployed in the development, test, and production environments, and also in the reporting environment; all other product modules are deployed only in the reporting environment.

Availability requirements

The deployment is divided into two distinct architectures:

- ▶ Development-test-production for DataStage
- ▶ Reporting for the Foundation Tools (IA, FT, BG, MWB)

Therefore, the availability mechanisms will be specific for each deployment architecture.

The business requirements indicate that a high volume of data will be processed in a relatively small window of time because of the international nature of the insurance company. Under these circumstances a grid architecture can provide the most flexibility and scalability to manageability.

In the production environment, the simplest topology is to co-locate the services and repository tiers along with the engine tier head node (including the resource manager) on a single machine (see Figure 2-20 on page 53). Depending on the number of required compute nodes, each compute node can reside on a separate machine. Additional nodes can easily be added as the volume of data processed increases over time.

Another advantage of the grid topology is that the same compute nodes that service the production environment during the window of time (usually during the night hours), can also service the development and test environments during the day to maximize use of the hardware, provide additional processing power to the

other environments, and provide a more accurate simulation of the production environment.

In the case of the insurance company, its development and test topologies are identical to the production environment (although they are likely to have different hardware specifications in terms of CPU cores and RAM). The reason is because there seems to be no justification for a more complex topology according to the business requirements. However, there is no reason why the tiers cannot be located on separate servers, configured for active-passive high availability, or clustered, and still share the grid (compute nodes) with the production environment. However, there is no reason that the grid must be shared across all environments.

Figure 6-6 shows how the three environments might share the grid compute nodes.

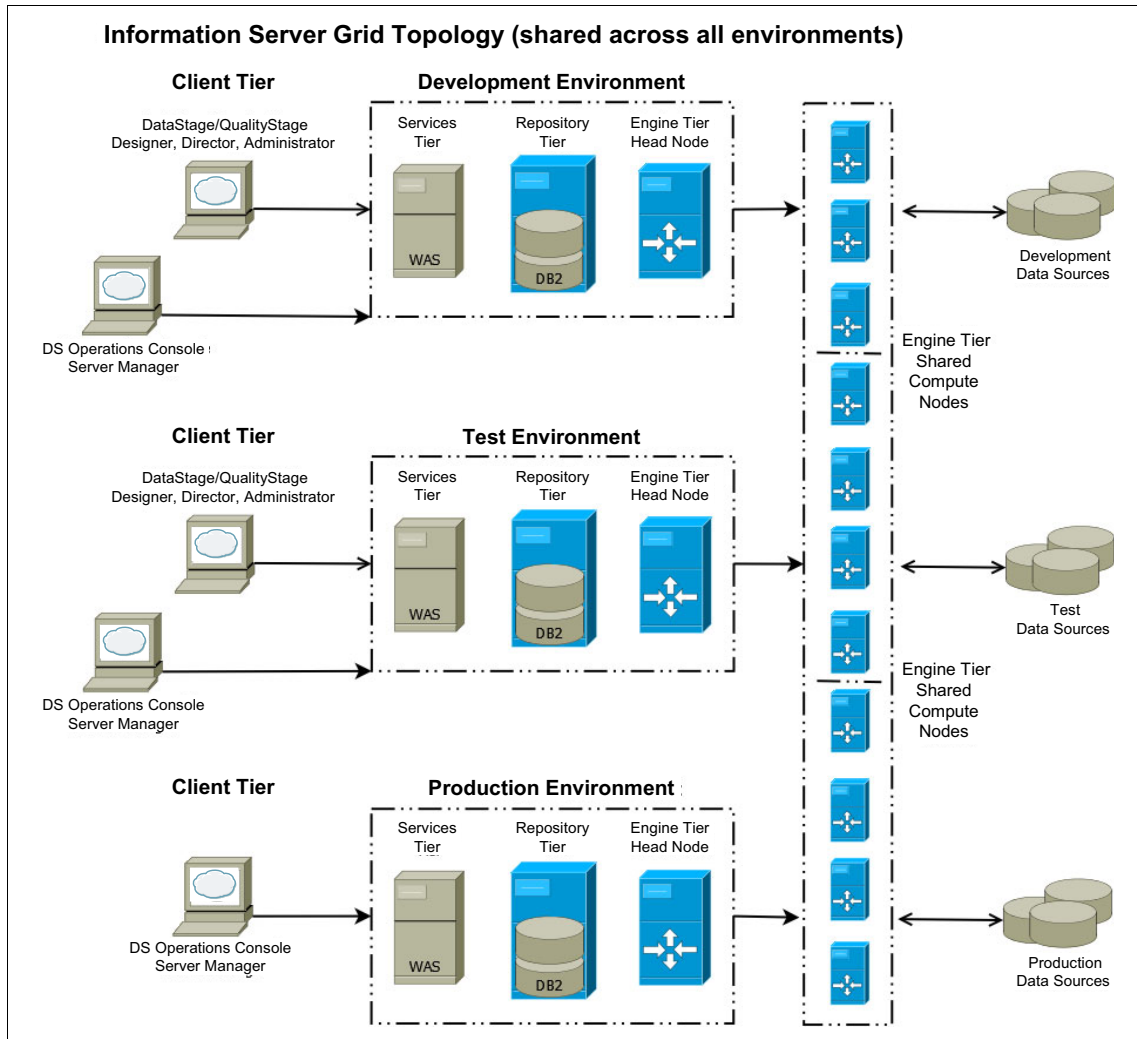


Figure 6-6 Shared grid topology for development, test, and production environments

The fourth environment will be the reporting environment. According to the business requirements, there will be a sophisticated business glossary with many relationships between terms and technical assets, and term-to-term relationships (such as terms that are developed by a specific company as related to the IIW reference glossary). In addition, the users will rely on data lineage reports, which sometimes require heavy processing on the services tier.

As a result, setting up a clustered services tier with two WebSphere Application Server nodes (to begin with) is suggested.

The only parallel engine requirements in this environment will be the data quality rules for monitoring and reporting through IA. Because execution will not be daily, the execution window of time can be flexible. Therefore, the engine tier can be co-located with the repository tier and not interfere with general operations. However, if in the future, loads on the engine negatively affect performance, a possibility is for this engine to share the compute nodes with the grid also.

It appears that the topology for the reporting environment will be similar to the UNIFIED environment from Scenario 2, except that Scenario 3 has only one engine. Figure 6-7 shows how the reporting environment topology might look.

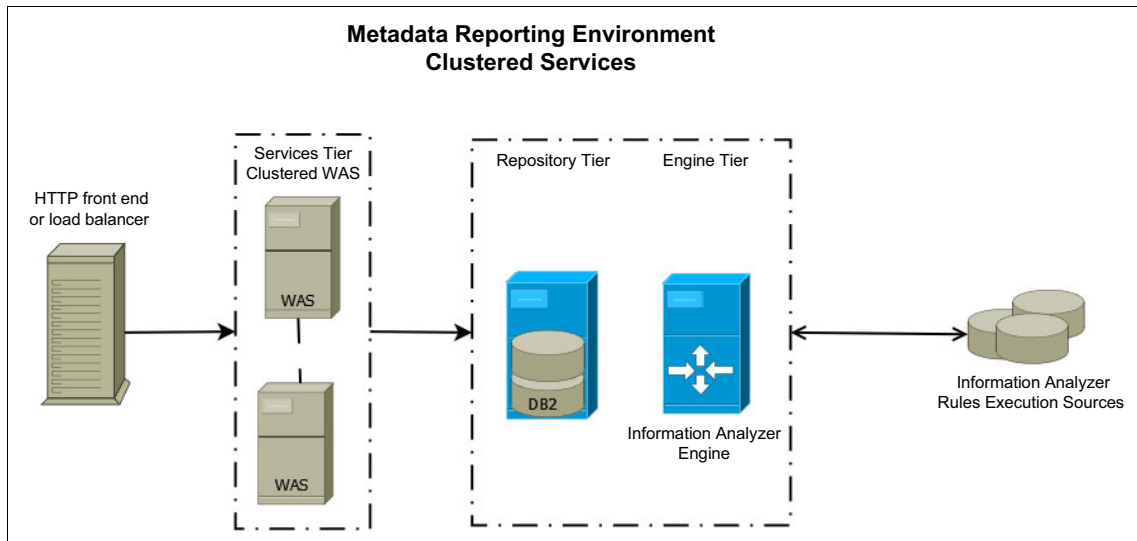


Figure 6-7 Metadata reporting environment with clustered WebSphere Application Server (services tier)

How to make it work

Again, the typical SDLC methodology that is described in Chapter 4, “Lifecycle management of assets” on page 75 (promoting DataStage jobs from development, to test, to production by using source control software and the InfoSphere Information Server Manager) can be used for the insurance company’s ETL processing with DataStage. In particular, the methodology is supportive of an information governance initiative.

A suggestion is for Metadata Workbench to be deployed both in the reporting instance and in the development instance. However, first the development environment is presented. Metadata Workbench uses the data source technical metadata, DataStage job design, and operational metadata from DataStage job runs. The technical metadata can be imported using InfoSphere Metadata Asset Manager or even DataStage itself. Because this environment is development, any method for loading the metadata is satisfactory (as long as it is stored in a consistent manner). The job design is already in the repository because that is where DataStage is persisted. Finally, the operational metadata is collected from running jobs in the Development environment, so no modification is required. All metadata that is used by Metadata Workbench is either imported into the development environment or generated and persisted there, so all Metadata Workbench reports should be available. See the Metadata Workbench user documentation for more details about generating impact analysis and data lineage reports:

http://publib.boulder.ibm.com/infocenter/iisinfsv/v8r7/topic/com.ibm.swg.im.iis.mdwb.admin.doc/topics/ct_administeringMetadataWorkbench.html

The focus in the remainder of this section is the reporting environment and the methodology for administering the metadata that is generated by the InfoSphere Information Server product modules.

We use the following steps:

1. The relevant external metadata must be imported into the reporting instance of InfoSphere Information Server. Using the InfoSphere Metadata Asset Manager and InfoSphere Data Architect metabridge-broker, import the InfoSphere Data Architect logical data (LDM) and physical data (DBM) models for the new data warehouse into the repository.

See the documentation for details about importing metadata with InfoSphere Metadata Asset Manager:

http://publib.boulder.ibm.com/infocenter/iisinfsv/v8r7/topic/com.ibm.swg.im.iis.mmi.doc/topics/t_importing_and_sharing.html

2. By using the InfoSphere Metadata Asset Manager connectors (where possible), import the metadata from all existing data sources that are currently in use in the production environment. This provides the bulk of the metadata

required to create the mapping specifications that will be used in the ETL design phase. The source metadata and the stored connections will be reused by IA in the process of connecting to the data sources for running the data quality rules metrics.

3. From the Business Glossary administration tab, import the IIW glossary that is included with the IBM Industry model. It includes relationships between the glossary terms and the imported physical model from DA. This semantic enrichment of the model can assist the data analysts who are doing the source-to-target mapping tasks.
4. Import any relevant business intelligence reports from Cognos into this repository also, by using the InfoSphere Metadata Asset Manager Cognos metabridge-broker. This metadata will be available for term assignment in Business Glossary, and for data lineage reporting from Metadata Workbench.

BI reports: The BI reports might not be available until a later stage in the project, however, importing the metadata should occur when the reports are available.

At the same time that the DataStage jobs are promoted to the production environment, the jobs should also be promoted to the reporting environment. Metadata Workbench uses the DataStage job design to help determine the data lineage and impact analysis results. This step (promoting DataStage code to the InfoSphere Information Server reporting instance) must be added to the standard SDLC procedure, because this task this will be on-going. Fortunately, promoting the DataStage code is needed in the reporting instance exactly when the code is promoted to production, so this step is straightforward.

Notes: To import DataStage jobs into an instance of InfoSphere Information Server, having an engine installed (that is used only by IA) is not enough. Therefore, installing DataStage in the reporting instance also is necessary, even though there will be no DataStage job execution.

Special licensing regulations exist regarding the installation of a “non-execute” instance of DataStage. Consult with your InfoSphere Information Server sales personnel for details.

When the DataStage jobs are run in the production environment, the operational metadata (OMD) should be collected from the *production* environment, but loaded into the *reporting* environment; loading it into the production environment is *not* necessary, because no other product modules will be using it there. The process for collecting and loading operational metadata in the production environment is described in 6.2.1, “Scenario 1: Simple” on page 198.

For Scenario 3, the process for collecting the operational metadata remains the same, however, there are two additional steps in the load phase.

1. Move the XML files where the operational metadata is stored from the production environment to the file system that is used by the engine tier in the reporting environment.
2. Edit the XML files: Change the name of the server that is hosting the engine tier from the production server host name to the reporting server host name before running the shell script that loads the operational metadata into the repository.

Figure 6-8 shows the XML tag in the operational metadata files where the engine host name is located for modification.

```
<LocatorComponent Class="Computer" Name="prodHost" />
<LocatorComponent Class="SoftwareProduct" Name="DataStage" />
```

Figure 6-8 Actual production Engine identifier tag in XML operational metadata files

Figure 6-9 shows the modified snippet that identifies the engine in the reporting environment.

```
<LocatorComponent Class="Computer" Name="reportHost" />
<LocatorComponent Class="SoftwareProduct" Name="DataStage" />
```

Figure 6-9 Modified XML tag value that identifies the report “engine”

Notes:

- ▶ The following tag is used frequently in the XML files:

```
<LocatorComponent Class="Computer" Name="prodHost" />
```

However, when it is used together with the following tag, you must change host name from <prodHost> to <reportHost>:

```
<LocatorComponent Class="SoftwareProduct" Name="DataStage" />
```

- ▶ Do not automatically do a search and replace operation for all instances of the prodHost name.
- ▶ There is also “Computer” Name=’ for other “SoftwareProduct” such as RDBMSs (DB2, Oracle, and Teradata, as examples) that identifies the host where the data sources reside (not the host where the engine resides).
However, only when the “SoftwareProduct” Name=“DataStage”’, should the host name identifier be changed to the reporting engine host name.

After modifying all of the XML files to include the reporting engine instead of the production engine, the operational metadata must be loaded into the repository of the reporting instance by using the `RunImportStart.sh` shell script. This modification is necessary to ensure that Metadata Workbench can associate the operational metadata with the DataStage job designs that were imported into DataStage projects on the engine tier of the reporting instance.

Although the DataStage jobs were developed in the development environment and executed in the production environment, importing the DataStage jobs attaches them to the local engine, and the operational metadata must match.

The reporting instance should now have all of the assets that the product modules require to perform their duties.

6.3 Summary

This book describes the infrastructure of InfoSphere Information Server, various architectural topologies, the product modules that comprise the product suite, their functionality and basic usage, and their integration. In addition, the SDLC and techniques for managing it were presented for DataStage and QualityStage, because they are the primary code development product modules in the InfoSphere Information Server product suite. However, the book also describes how metadata forms a major part of the integration between the product modules. Metadata is not only a by-product of using the various product modules, but metadata is as much a deliverable itself, as is the data from the ETL

processes. Therefore, the lifecycle of metadata is introduced as a factor that differs significantly from the SDLC of DataStage and QualityStage, and must be managed differently too.

To successfully deliver both the resulting data (from the ETL processes) for business intelligence, data warehouse, master data management, and other purposes, and the resulting metadata (from the product module generation and importation) for governance, impact, lineage and other purposes, this book presents several deployment strategies and techniques for determining an appropriate deployment architecture to achieve the business objectives. These strategies consider product mix, user population mix, usage behavior, reporting requirements, availability requirements, and load and stress on the various components of the InfoSphere Information Server infrastructure by the product modules.

By understanding the underlying principles of InfoSphere Information Server that this book presents, and the specific analyses of the examples in this chapter, you should be able to determine which InfoSphere Information Server product modules will fulfill business objectives, how to deploy these product modules in an optimal way to achieve maximum benefit, and the type of architectural topology that can ensure successful delivery of the designated InfoSphere Information Server products.

Glossary

analysis database A database that InfoSphere Information Analyzer uses when it runs analysis jobs and where it stores the extended analysis information. The analysis database is sometimes referred to by the term IADB. The analysis database does not contain the InfoSphere Information Analyzer projects, analysis results, and design-time information; all of this information is stored in the InfoSphere Information Server repository.

baseline analysis A type of data analysis that compares a saved view of the results of a column analysis to another view of the same results that are captured later.

business analyst A specialist who analyzes business needs and problems, consults with users and stakeholders to identify opportunities for improving business return through information technology, and transforms requirements into a technical form.

bridge A component that converts metadata from one format to another format by mapping the metadata elements to a standard model. This model translates the semantics of the source tool into the semantics of the target tool. For example, the source tool might be a business intelligence or data modeling tool, and the target tool might be the InfoSphere Information Server repository. Or, the source tool might be the InfoSphere Information Server repository, and the target tool might be a file that is used by a data modeling tool. See also *connector* and *InfoSphere Information Server repository*.

business intelligence assets Information assets that are used by business intelligence (BI) tools to organize reports and models that provide a business view of data. These assets include BI reports, BI models, BI collections, and cubes.

cardinality A measure of the number of unique values in a column.

Citrix Systems A company that works in partnership with IBM, and provide desktop virtualization, server virtualization, network optimization and collaboration products to help organizations create simpler and more cost-effective IT environments.

client tier The client programs and consoles that are used for development, administration, and other tasks for the InfoSphere Information Server suite and product modules and the computers where they are installed. definition.

column analysis A data quality process that describes the condition of data at the field level. You perform column analysis by defining and running a column analysis job and then examining the results.

common domain Columns share a common domain when they share the same values.

compute node A processing node in a parallel processing environment that handles elements of the job logic. Any processing node that is not a conductor node is a compute node. Each engine tier can have one or more compute nodes. See also *processing node* and *conductor node*.

conductor node The processing node that initiates the job run. Each engine tier has only one conductor node. See also *processing node*.

connector A component that provides data connectivity and metadata integration for external data sources, such as relational databases or messaging software. A connector typically includes a stage that is specific to the external data source. See also *bridge*.

constant Data that has an unchanging, predefined value to be used in processing.

cross-domain analysis A type of analysis that identifies the overlap of data values between two columns of data.

cross-table analysis A type of analysis that combines foreign key analysis and cross-domain analysis.

data analyst A technical expert who creates, runs, and reviews data analysis processes.

data class A classification that designates the logical type of data in a data field. For example, the INDICATOR classification represents a binary value such as TRUE/FALSE or YES/NO.

data field A column or field that contains a specific set of data values common to all records in a file or table.

data file (1) A file that stores a collection of fields (which is called a data file structure) in a native system file instead of in a database table.
(2) The information asset that represents a collection of fields stored in a single file, such as a flat file (a file that has no hierarchical structure), a complex flat file (a file that has hierarchical structure, especially mainframe data structures and XML files), or a sequential file.

data profiling Understanding the source data quality and structure.

data rule An expression generated out of a data rule definition that evaluates and analyzes conditions found during data profiling and data quality assessment.

data rule definition A true or false (Boolean) expression that contains various conditions to evaluate data records.

data set A set of parallel data files and the descriptor file that refers to them. Data sets optimize the writing of data to disk by preserving the degree of partitioning. See also *data file*.

data source The source of data itself, such as a database or XML file, and the connection information necessary for accessing the data.

data subclass A user-defined subclass that is used to designate more specifically the type of data. For example, within the Quantity class category, the subclass 01 might stand for a data field that holds information about currency. The 01 subclass category might mean 1 dollar in this context.

disaster recovery The ability to recover a data center at a different site, on different hardware, if a disaster destroys the entire primary site or renders it inoperable.

domain The set of data in a column.

domain layer See *services tier*.

DS engine See *InfoSphere Information Server engine* and *server engine*.

engine See *InfoSphere Information Server engine*.

engine tier The logical group of engine components for the InfoSphere Information Server suite and product modules (the InfoSphere Information Server engine components, service agents, and so on) and the computer or computers where those components are installed.

foreign key analysis A type of analysis that is part of the key and relationship functionality. This analysis discovers cross-table relationships by identifying foreign key candidates in table data. Foreign keys reference primary keys that are already defined or identified during primary key analysis.

format frequency The frequency distribution of general formats that are used in a column.

frequency distribution The number of occurrences of each unique value in a column.

front-end node See *conductor node*.

general format The use of a character symbol for each unique data value. For example, all alphabetic characters in a column are replaced with the letter A.

global logical variable A value that you set to represent a fact or a specific piece of data. It is a shared construct that can be used in all of your data rule definitions and data rules.

head node See *conductor node*.

high availability The systems that are able to recover quickly when hardware failures occur.

inferences A statistical inference in which probabilities are interpreted as degrees of belief.

inferred data type When values in a column are examined during a column analysis job, the values are examined for different data types. From the list of inferred data types, a recommendation is made as to the best data type to present during the review process.

InfoSphere Information Server repository A shared component that stores design-time, run time, glossary, and other metadata for product modules in the InfoSphere Information Server suite. The Information Server repository is installed in the Information Server repository tier.

InfoSphere Information Server repository tier The InfoSphere Information Server repository and, if installed, the InfoSphere Information Analyzer database (analysis database) and the computer or computers where these components are installed.

InfoSphere Information Server engine The software that runs tasks or jobs such as discovery, analysis, cleansing, or transformation. The engine includes the server engine, the parallel engine, and the other components that make up the runtime environment for InfoSphere Information Server and its product modules.

InfoSphere Information Server instance The installed set of components that are part of the engine tier, services tier, and InfoSphere Information Server repository tier.

input link A link that connects a data source to a stage. See also *link*.

job The design objects and compiled programmatic elements that can connect to data sources, extract and transform that data, and then can load that data into a target system. Types of jobs include parallel jobs, sequence jobs, server jobs, and mainframe jobs. See also *job design* and *job executable*.

job design The metadata that defines the sources and targets that are used within a job and the logic that operates on the associated data. A job design is composed of stages and the links between those stages. Developers use visual tools to create job designs. The job design is stored in the InfoSphere Information Server repository, separate from the job executable. See also *job*, *job executable*, *stage*, and *link*.

job executable The set of binary objects, generated scripts, and associated files that are used when running a job. The job executable is stored in the projects directory on the engine tier. See also *job* and *job design*.

job parameter A processing variable that can be used at various points in a job design and overridden when the job is executed to dynamically influence the processing of the job. Job parameters are most often used for paths, file names, database connection information, or job logic. See also *job* and *job design*.

job run A specific run of a job. A job can run multiple times, producing multiple job runs.

job sequence See *sequence job*.

layer See *tier*.

link A representation of a data flow that joins the stages in a job. A link connects data sources to processing stages, connects processing stages to each other, and also connects those processing stages to target systems. The types of links are input link, output link, reference link, and reject link.

lookup table (1) A data source that has a key value that jobs use to retrieve reference information. (2) A database table that is used to map one or more input values to one or more output values.

master schema definition A physical model of the inferred properties that is generated out of the data selected. It reflects the inferences of the data instead of the original definitions of the metadata.

Match Designer database A database that stores the results of match test passes that are generated by InfoSphere QualityStage

metadata Data that describes the characteristics of data.

metadata services A shared set of components that provide common functions (such as import and export) to other product modules in the InfoSphere Information Server suite and that are installed on all tiers.

metric An equation that you define to develop a measurement you can apply against data rules, rule sets, and other metrics

natural key analysis A type of analysis that is part of the key and relationship functionality. This analysis evaluates data tables to find natural key candidates.

node A logical processing unit that is defined in a configuration file by a virtual name and a set of associated details about the physical resources, such as the server, the disks, its pools, and so on.

nullability An attribute that determines whether null values are allowed.

operational metadata Metadata that describes the events and processes that occur and the objects that are affected when you run a job.

operations database A component of the InfoSphere Information Server repository that stores both the operational metadata and the information about the system resources that were used when a job is run for the product modules in the InfoSphere Information Server suite. See also *operational metadata* and *InfoSphere Information Server repository*.

operator A runtime object library that is part of the parallel engine and that executes the logic as defined in its corresponding stage. See also *stage*, *connector*, and *job executable*.

output link A link that is connected to a stage and generally moves processed data from the stage. See also *link*.

pack A collection of components that extends IBM InfoSphere DataStage capabilities.

parallel job A job that is compiled and run on the parallel engine and that supports parallel processing system features, including data pipelining, partitioning, and distributed execution. See also *Job*.

parallel engine The component of the InfoSphere Information Server engine that runs parallel jobs. Previously referred to as *PX engine*.

plug-in A type of stage that is used to connect to data sources but that does not support parallel processing capabilities. See also *connector*.

primary key analysis A type of analysis that is part of the key and relationship functionality. This analysis evaluates data tables to find primary key candidates.

processing node The logical nodes in the system where jobs are run. The configuration file can define one processing node for each physical node in the system or multiple processing nodes for each physical node.

project A container that organizes and provides security for objects that are supplied, created, or maintained for data integration, data profiling, quality monitoring, and so on.

PX engine See *parallel engine*.

reference link An input link on a Transformer or Lookup stage that defines where the lookup tables exist. See also *link*.

reference table A data table that you create or use for data analysis.

referential integrity A type of analysis that is run after foreign key analysis to ensure that foreign key candidates match the values of an associated primary key.

reject link An output link that identifies errors when the stage is processing records and that routes those rejected records to a target stage. See also *link* and *output link*.

relationship A defined connection between the rows of a table or the rows of two tables. A relationship is the internal representation of a referential constraint.

repository A persistent storage area for data and other application resources.

review The process of evaluating data analysis results. During the review, you accept or reject inferences made during analysis.

rule set definition A collection of data rule definitions.

sequence job A job that runs other jobs in a specified order.

server engine The component of the InfoSphere Information Server engine that runs server jobs and job sequences. Previously referred to as *DS engine*.

server job A job that is compiled and run on the server engine.

services tier The application server, common services, and product services for the InfoSphere Information Server suite and product modules and the computer or computers where those components are installed.

stage The element of a job design that describes a data source, a data processing step, or a target system and that defines the processing logic that moves data from input links to output links. A stage is a configured instance of a stage type. See also *stage type* and *job design*.

stage type An object that defines the capabilities of a stage, the parameters of the stage, and the libraries that the stage uses at run time. See also *stage*.

system inference See *inferences*

table analysis An analysis process that consists of primary key analysis and the assessment of multicolumn primary keys and potential duplicate values.

tier The logical group of components and the computers on which those components are installed. Previously referred to as *layer*.

trigger A representation of dependencies between workflow tasks that joins activities in a sequence job. Activities typically have one input trigger, but multiple output triggers. See also *sequence job*.

uniqueness threshold A column analysis setting that infers whether a column is unique.

user classification code A user-defined code that is assigned to a data field. For example, the code FIN might refer to financial data.

validity assessment A data analysis process that evaluates data columns for valid and invalid values.

virtual column A concatenation of two or more columns that can be analyzed as if it were an existing column.

Abbreviations and acronyms

ACR	automatic client reroute	LSF	Load Sharing Facility
API	application programming interface	MPP	massively parallel processing
BI	business intelligence	MS	Microsoft (abbreviation is used in the figures of this book)
CLI	command-line interface	MWB	Metadata Workbench
CSV	comma-separated value	NAS	network-attached storage
DEV	development	ODBC	Open Database Connectivity
DNS	Domain Name System	PRE-PROD	preproduction
DR	disaster recovery	PROD	production
DS	DataStage (abbreviation is used in the figures of this book)	PVU	processor value unit
DSN	data source name	OGE	Oracle Grid Engine
ETL	extract, transform, and load	QS	QualityStage (abbreviation is used in the figures of this book)
GFS	Global File System	QSDB	QualityStage database
GPFS	General Parallel File System	RAC	Real Application Clusters
GUI	graphical user interface	RAM	random access memory
HADR	high availability disaster recovery	RDBMS	relational database management system
IADB	information analyzer database	REST	Representational State Transfer
IDE	integrated development environment	SAN	storage area network
IP	Internet Protocol	SGE	Sun Grid Engine
IS	InfoSphere Information Server (abbreviation is used in the figures of this book)	SMP	symmetric multiprocessor
ISD	InfoSphere Services Director	SQL	Structured Query Language
ITSO	International Technical Support Organization	SSH	Secure Shell
J2EE	Java 2 Enterprise Edition	URL	Uniform Resource Locator
JVM	Java virtual machine	WLM	Workload Manager
LDAP	Lightweight Directory Access Protocol	XML	Extensible Markup Language

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this book.

IBM Redbooks

The following IBM Redbooks publications provide additional information about the topic in this document. Note that some publications referenced in this list might be available in softcopy only.

- ▶ *Information Server: Installation and Configuration Guide*, REDP-4596
- ▶ *Metadata Management with IBM InfoSphere Information Server*, SG24-7939

You can search for, view, download or order these documents and other Redbooks, Redpapers, Web Docs, draft and additional materials, at the following website:

ibm.com/redbooks

Online resources

The following website is also relevant for terms and definitions from IBM software and hardware products:

<http://www.ibm.com/software/globalization/terminology/>

Help from IBM

IBM Support and downloads

ibm.com/support

IBM Global Services

ibm.com/services



IBM InfoSphere Information Server Deployment Architectures

(0.5" spine)
0.475" <-> 0.875"
250 <-> 459 pages



IBM InfoSphere Information Server Deployment Architectures



Start small with a path for massive growth

Understand metadata considerations

Deploy in Grid and SMP environments

Typical deployment architectures introduce challenges to fully using the shared metadata platform across products, environments, and servers. Data privacy and information security requirements add even more levels of complexity. IBM InfoSphere Information Server provides a comprehensive, metadata-driven platform for delivering trusted information across heterogeneous systems.

This IBM Redbooks publication presents guidelines and criteria for the successful deployment of InfoSphere Information Server components in typical logical infrastructure topologies that use shared metadata capabilities of the platform, and support development lifecycle, data privacy, information security, high availability, and performance requirements. This book can help you evaluate information requirements to determine an appropriate deployment architecture, based on guidelines that are presented here, and that can fulfill specific use cases. It can also help you effectively use the functionality of your Information Server product modules and components to successfully achieve your business goals.

This book is for IT architects, information management and integration specialists, and system administrators who are responsible for delivering the full suite of information integration capabilities of InfoSphere Information Server.

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks